



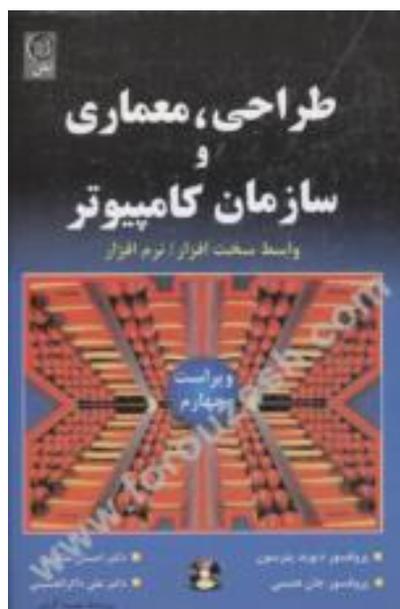
دانشگاه آزاد اسلامی واحد بافت

خلاصه کتاب طراحی، معماری

و

سازمان کامپیوتر

واسط سخت افزار / نرم افزار



تهیه کننده :

کاظم فریدی

دانشجوی کارشناسی ارشد مهندسی معماری سیستمهای کامپیوتری

با سلام

دوستان عزیز موارد خلاصه شده فوق چیزهایی است که استاد گرامی در کلاسهایی که برگزار گردید توسط بنده برداشت شده است و این خلاصه نویسی فقط برای راهنمایی استفاده شما در خواندن مطالب کتاب می باشد اما در صورت داشتن وقت سعی نمایید کلیه مطالب فصول کتاب را برای فهم بیشتر خوانده و فراگیرید . با تشکر فریدی

**فصول یک ، دو و سه توضیحاتی برای فراگیری و یادآوری دروس گذشته دوره لیسانس**  
**میباشد که صفحات قابل خواندن و درک کردن مشخص گردیده که میتوانید برای فراگیری و یادآوری دوباره حتما مطالعه نمایید .**

### خلاصه فصل اول

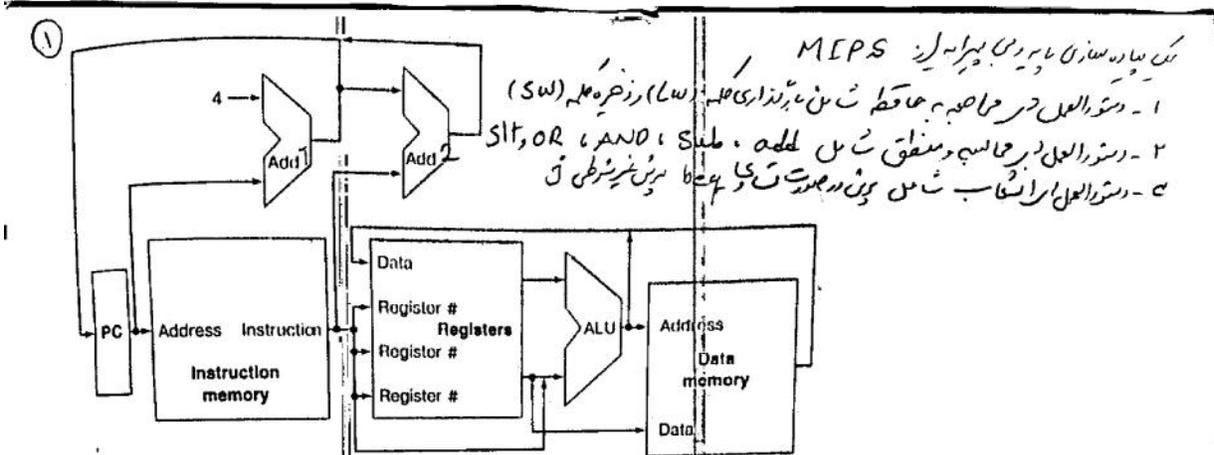
از صفحه ۱۹: (رده های گوناگون کاربردهای کامپیوی و شاخص های آنها ) تا صفحه ۲۱  
 از صفحه ۲۴: (۱-۲ فراسوی برنامه های شما از زبان سطح بالا به زبان سخت افزار ) تا پایان صفحه ۲۷  
 صفحه ۲۸: دورنما  
 صفحه ۲۹ شکل ۱-۴  
 صفحه ۳۱ پاراگراف دوم (هر تصویر متشکل از ماتریسی ....)  
 صفحه ۳۳ پاراگراف دوم (حافظه مکانی برای ذخیره تا پایان پاراگراف)  
 صفحه ۳۴ از شکل ۱-۹ تا پایان صفحه  
 صفحه ۳۵ دورنما  
 صفحه ۳۷ پاراگراف دوم تا آخر پاراگراف (با این اوصاف ....)

### خلاصه فصل دوم

از صفحه ( ۱۰۹ فیلدهای دستورالعمل MIPS ) تا پایان صفحه ۱۱۲ دورنما  
 صفحه ۱۱۶ دستورالعمل های تصمیم گیری تا حلقه ها صفحه ۱۱۸  
 صفحه ۱۴۳ جمع بندی سبک های آدرس دهی در MIPS تا بالای صفحه ۱۴۴

### فصل سوم هیچ موردی تدریس نگردید

**فصل چهارم و پنجم و هفتم نیز بصورت خلاصه شده با توضیحات اشکال در زیر برای شما اسکن شده**  
**است که امیدوارم مورد استفاده شما دوستان گرامی قرار بگیرد و بتوانید نمره خوبی را در امتحان**  
**پایان ترم بدست آورید .**



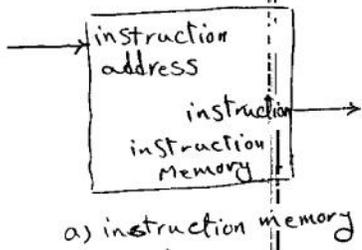
نویسندگان سازنده این پردازنده MIPS  
 ۱- دستور العمل هر محاسبه به حافظه است منبذاری (Sw) (Lw) ذخیره عمل (Sw)  
 ۲- دستور العمل در محاسبه و منطق است من جمله: AND, OR, SLT, OR  
 ۳- دستور العمل برای انتخاب است من برون در صورتی که در ۴ یا ۵ برون غیر شرطی

در پیاده سازی هر دستور العمل ۲ کامپلکسیت است: ۱- باید مقدار کد دستور برای PC به حافظه که کد برنامه را در ذخیره های داده ارسال کرد  
 و پس از آن نقطه از حافظه، دستور العمل که ظاهر واکس Test کد کرد.  
 ۲- بر اساس فیلدهای دستور العمل واکس شده باسی است تا آنجا که دستور العمل انتخاب و خوانده شود. (دستور العمل با کد  
 که در حافظه است) بنابراین کد است با در انتخاب و خوانده شود در حافظه که در بیشتر دستور العمل در کد برنامه به خواندن در کد نیاز مند  
 است: تمام دستور العمل در هر سه دره (به استثنای دستور العمل برون غیر شرطی) منبذاری خواندن منبذاری است که با علامه از فیلدهای منبذاری  
 (ALU) استفاده می کنند. دستور العمل در رجوع به حافظه برای عمل آمدن از ALU به منبذاری  
 دستور العمل در رجوع به حافظه: برای خواندن داده از حافظه یا نوشتن داده در آن نیاز مند دسترسی (access) به حافظه  
 دستور العمل در رجوع به منطق، دستور العمل با در رجوع به ALU، خودی حافظه ظاهر شده را در کد است  
 ماژولوسی کنند و در آخر دستور العمل برای انتخاب (بروش شرطی) منبذاری است بر اساس نتیجه منبذاری که ALU انجام داده خواهد آمد  
 دستور العمل بعد از آن تغییر می دهند و با آنکه ماژولوسی کد را در PC (نویسندگان) آمدن دستور العمل پس از آنکه

تفسیر عمل:  
 مقدار که در PC نوشته شود می تواند از مقدار کد که در رجوع است. ۱، ۲، ۳، ۴، ۵، ۶، ۷، ۸، ۹، ۱۰، ۱۱، ۱۲، ۱۳، ۱۴، ۱۵، ۱۶، ۱۷، ۱۸، ۱۹، ۲۰، ۲۱، ۲۲، ۲۳، ۲۴، ۲۵، ۲۶، ۲۷، ۲۸، ۲۹، ۳۰، ۳۱، ۳۲، ۳۳، ۳۴، ۳۵، ۳۶، ۳۷، ۳۸، ۳۹، ۴۰، ۴۱، ۴۲، ۴۳، ۴۴، ۴۵، ۴۶، ۴۷، ۴۸، ۴۹، ۵۰، ۵۱، ۵۲، ۵۳، ۵۴، ۵۵، ۵۶، ۵۷، ۵۸، ۵۹، ۶۰، ۶۱، ۶۲، ۶۳، ۶۴، ۶۵، ۶۶، ۶۷، ۶۸، ۶۹، ۷۰، ۷۱، ۷۲، ۷۳، ۷۴، ۷۵، ۷۶، ۷۷، ۷۸، ۷۹، ۸۰، ۸۱، ۸۲، ۸۳، ۸۴، ۸۵، ۸۶، ۸۷، ۸۸، ۸۹، ۹۰، ۹۱، ۹۲، ۹۳، ۹۴، ۹۵، ۹۶، ۹۷، ۹۸، ۹۹، ۱۰۰، ۱۰۱، ۱۰۲، ۱۰۳، ۱۰۴، ۱۰۵، ۱۰۶، ۱۰۷، ۱۰۸، ۱۰۹، ۱۱۰، ۱۱۱، ۱۱۲، ۱۱۳، ۱۱۴، ۱۱۵، ۱۱۶، ۱۱۷، ۱۱۸، ۱۱۹، ۱۲۰، ۱۲۱، ۱۲۲، ۱۲۳، ۱۲۴، ۱۲۵، ۱۲۶، ۱۲۷، ۱۲۸، ۱۲۹، ۱۳۰، ۱۳۱، ۱۳۲، ۱۳۳، ۱۳۴، ۱۳۵، ۱۳۶، ۱۳۷، ۱۳۸، ۱۳۹، ۱۴۰، ۱۴۱، ۱۴۲، ۱۴۳، ۱۴۴، ۱۴۵، ۱۴۶، ۱۴۷، ۱۴۸، ۱۴۹، ۱۵۰، ۱۵۱، ۱۵۲، ۱۵۳، ۱۵۴، ۱۵۵، ۱۵۶، ۱۵۷، ۱۵۸، ۱۵۹، ۱۶۰، ۱۶۱، ۱۶۲، ۱۶۳، ۱۶۴، ۱۶۵، ۱۶۶، ۱۶۷، ۱۶۸، ۱۶۹، ۱۷۰، ۱۷۱، ۱۷۲، ۱۷۳، ۱۷۴، ۱۷۵، ۱۷۶، ۱۷۷، ۱۷۸، ۱۷۹، ۱۸۰، ۱۸۱، ۱۸۲، ۱۸۳، ۱۸۴، ۱۸۵، ۱۸۶، ۱۸۷، ۱۸۸، ۱۸۹، ۱۹۰، ۱۹۱، ۱۹۲، ۱۹۳، ۱۹۴، ۱۹۵، ۱۹۶، ۱۹۷، ۱۹۸، ۱۹۹، ۲۰۰، ۲۰۱، ۲۰۲، ۲۰۳، ۲۰۴، ۲۰۵، ۲۰۶، ۲۰۷، ۲۰۸، ۲۰۹، ۲۱۰، ۲۱۱، ۲۱۲، ۲۱۳، ۲۱۴، ۲۱۵، ۲۱۶، ۲۱۷، ۲۱۸، ۲۱۹، ۲۲۰، ۲۲۱، ۲۲۲، ۲۲۳، ۲۲۴، ۲۲۵، ۲۲۶، ۲۲۷، ۲۲۸، ۲۲۹، ۲۳۰، ۲۳۱، ۲۳۲، ۲۳۳، ۲۳۴، ۲۳۵، ۲۳۶، ۲۳۷، ۲۳۸، ۲۳۹، ۲۴۰، ۲۴۱، ۲۴۲، ۲۴۳، ۲۴۴، ۲۴۵، ۲۴۶، ۲۴۷، ۲۴۸، ۲۴۹، ۲۵۰، ۲۵۱، ۲۵۲، ۲۵۳، ۲۵۴، ۲۵۵، ۲۵۶، ۲۵۷، ۲۵۸، ۲۵۹، ۲۶۰، ۲۶۱، ۲۶۲، ۲۶۳، ۲۶۴، ۲۶۵، ۲۶۶، ۲۶۷، ۲۶۸، ۲۶۹، ۲۷۰، ۲۷۱، ۲۷۲، ۲۷۳، ۲۷۴، ۲۷۵، ۲۷۶، ۲۷۷، ۲۷۸، ۲۷۹، ۲۸۰، ۲۸۱، ۲۸۲، ۲۸۳، ۲۸۴، ۲۸۵، ۲۸۶، ۲۸۷، ۲۸۸، ۲۸۹، ۲۹۰، ۲۹۱، ۲۹۲، ۲۹۳، ۲۹۴، ۲۹۵، ۲۹۶، ۲۹۷، ۲۹۸، ۲۹۹، ۳۰۰، ۳۰۱، ۳۰۲، ۳۰۳، ۳۰۴، ۳۰۵، ۳۰۶، ۳۰۷، ۳۰۸، ۳۰۹، ۳۱۰، ۳۱۱، ۳۱۲، ۳۱۳، ۳۱۴، ۳۱۵، ۳۱۶، ۳۱۷، ۳۱۸، ۳۱۹، ۳۲۰، ۳۲۱، ۳۲۲، ۳۲۳، ۳۲۴، ۳۲۵، ۳۲۶، ۳۲۷، ۳۲۸، ۳۲۹، ۳۳۰، ۳۳۱، ۳۳۲، ۳۳۳، ۳۳۴، ۳۳۵، ۳۳۶، ۳۳۷، ۳۳۸، ۳۳۹، ۳۴۰، ۳۴۱، ۳۴۲، ۳۴۳، ۳۴۴، ۳۴۵، ۳۴۶، ۳۴۷، ۳۴۸، ۳۴۹، ۳۵۰، ۳۵۱، ۳۵۲، ۳۵۳، ۳۵۴، ۳۵۵، ۳۵۶، ۳۵۷، ۳۵۸، ۳۵۹، ۳۶۰، ۳۶۱، ۳۶۲، ۳۶۳، ۳۶۴، ۳۶۵، ۳۶۶، ۳۶۷، ۳۶۸، ۳۶۹، ۳۷۰، ۳۷۱، ۳۷۲، ۳۷۳، ۳۷۴، ۳۷۵، ۳۷۶، ۳۷۷، ۳۷۸، ۳۷۹، ۳۸۰، ۳۸۱، ۳۸۲، ۳۸۳، ۳۸۴، ۳۸۵، ۳۸۶، ۳۸۷، ۳۸۸، ۳۸۹، ۳۹۰، ۳۹۱، ۳۹۲، ۳۹۳، ۳۹۴، ۳۹۵، ۳۹۶، ۳۹۷، ۳۹۸، ۳۹۹، ۴۰۰، ۴۰۱، ۴۰۲، ۴۰۳، ۴۰۴، ۴۰۵، ۴۰۶، ۴۰۷، ۴۰۸، ۴۰۹، ۴۱۰، ۴۱۱، ۴۱۲، ۴۱۳، ۴۱۴، ۴۱۵، ۴۱۶، ۴۱۷، ۴۱۸، ۴۱۹، ۴۲۰، ۴۲۱، ۴۲۲، ۴۲۳، ۴۲۴، ۴۲۵، ۴۲۶، ۴۲۷، ۴۲۸، ۴۲۹، ۴۳۰، ۴۳۱، ۴۳۲، ۴۳۳، ۴۳۴، ۴۳۵، ۴۳۶، ۴۳۷، ۴۳۸، ۴۳۹، ۴۴۰، ۴۴۱، ۴۴۲، ۴۴۳، ۴۴۴، ۴۴۵، ۴۴۶، ۴۴۷، ۴۴۸، ۴۴۹، ۴۵۰، ۴۵۱، ۴۵۲، ۴۵۳، ۴۵۴، ۴۵۵، ۴۵۶، ۴۵۷، ۴۵۸، ۴۵۹، ۴۶۰، ۴۶۱، ۴۶۲، ۴۶۳، ۴۶۴، ۴۶۵، ۴۶۶، ۴۶۷، ۴۶۸، ۴۶۹، ۴۷۰، ۴۷۱، ۴۷۲، ۴۷۳، ۴۷۴، ۴۷۵، ۴۷۶، ۴۷۷، ۴۷۸، ۴۷۹، ۴۸۰، ۴۸۱، ۴۸۲، ۴۸۳، ۴۸۴، ۴۸۵، ۴۸۶، ۴۸۷، ۴۸۸، ۴۸۹، ۴۹۰، ۴۹۱، ۴۹۲، ۴۹۳، ۴۹۴، ۴۹۵، ۴۹۶، ۴۹۷، ۴۹۸، ۴۹۹، ۵۰۰، ۵۰۱، ۵۰۲، ۵۰۳، ۵۰۴، ۵۰۵، ۵۰۶، ۵۰۷، ۵۰۸، ۵۰۹، ۵۱۰، ۵۱۱، ۵۱۲، ۵۱۳، ۵۱۴، ۵۱۵، ۵۱۶، ۵۱۷، ۵۱۸، ۵۱۹، ۵۲۰، ۵۲۱، ۵۲۲، ۵۲۳، ۵۲۴، ۵۲۵، ۵۲۶، ۵۲۷، ۵۲۸، ۵۲۹، ۵۳۰، ۵۳۱، ۵۳۲، ۵۳۳، ۵۳۴، ۵۳۵، ۵۳۶، ۵۳۷، ۵۳۸، ۵۳۹، ۵۴۰، ۵۴۱، ۵۴۲، ۵۴۳، ۵۴۴، ۵۴۵، ۵۴۶، ۵۴۷، ۵۴۸، ۵۴۹، ۵۵۰، ۵۵۱، ۵۵۲، ۵۵۳، ۵۵۴، ۵۵۵، ۵۵۶، ۵۵۷، ۵۵۸، ۵۵۹، ۵۶۰، ۵۶۱، ۵۶۲، ۵۶۳، ۵۶۴، ۵۶۵، ۵۶۶، ۵۶۷، ۵۶۸، ۵۶۹، ۵۷۰، ۵۷۱، ۵۷۲، ۵۷۳، ۵۷۴، ۵۷۵، ۵۷۶، ۵۷۷، ۵۷۸، ۵۷۹، ۵۸۰، ۵۸۱، ۵۸۲، ۵۸۳، ۵۸۴، ۵۸۵، ۵۸۶، ۵۸۷، ۵۸۸، ۵۸۹، ۵۹۰، ۵۹۱، ۵۹۲، ۵۹۳، ۵۹۴، ۵۹۵، ۵۹۶، ۵۹۷، ۵۹۸، ۵۹۹، ۶۰۰، ۶۰۱، ۶۰۲، ۶۰۳، ۶۰۴، ۶۰۵، ۶۰۶، ۶۰۷، ۶۰۸، ۶۰۹، ۶۱۰، ۶۱۱، ۶۱۲، ۶۱۳، ۶۱۴، ۶۱۵، ۶۱۶، ۶۱۷، ۶۱۸، ۶۱۹، ۶۲۰، ۶۲۱، ۶۲۲، ۶۲۳، ۶۲۴، ۶۲۵، ۶۲۶، ۶۲۷، ۶۲۸، ۶۲۹، ۶۳۰، ۶۳۱، ۶۳۲، ۶۳۳، ۶۳۴، ۶۳۵، ۶۳۶، ۶۳۷، ۶۳۸، ۶۳۹، ۶۴۰، ۶۴۱، ۶۴۲، ۶۴۳، ۶۴۴، ۶۴۵، ۶۴۶، ۶۴۷، ۶۴۸، ۶۴۹، ۶۵۰، ۶۵۱، ۶۵۲، ۶۵۳، ۶۵۴، ۶۵۵، ۶۵۶، ۶۵۷، ۶۵۸، ۶۵۹، ۶۶۰، ۶۶۱، ۶۶۲، ۶۶۳، ۶۶۴، ۶۶۵، ۶۶۶، ۶۶۷، ۶۶۸، ۶۶۹، ۶۷۰، ۶۷۱، ۶۷۲، ۶۷۳، ۶۷۴، ۶۷۵، ۶۷۶، ۶۷۷، ۶۷۸، ۶۷۹، ۶۸۰، ۶۸۱، ۶۸۲، ۶۸۳، ۶۸۴، ۶۸۵، ۶۸۶، ۶۸۷، ۶۸۸، ۶۸۹، ۶۹۰، ۶۹۱، ۶۹۲، ۶۹۳، ۶۹۴، ۶۹۵، ۶۹۶، ۶۹۷، ۶۹۸، ۶۹۹، ۷۰۰، ۷۰۱، ۷۰۲، ۷۰۳، ۷۰۴، ۷۰۵، ۷۰۶، ۷۰۷، ۷۰۸، ۷۰۹، ۷۱۰، ۷۱۱، ۷۱۲، ۷۱۳، ۷۱۴، ۷۱۵، ۷۱۶، ۷۱۷، ۷۱۸، ۷۱۹، ۷۲۰، ۷۲۱، ۷۲۲، ۷۲۳، ۷۲۴، ۷۲۵، ۷۲۶، ۷۲۷، ۷۲۸، ۷۲۹، ۷۳۰، ۷۳۱، ۷۳۲، ۷۳۳، ۷۳۴، ۷۳۵، ۷۳۶، ۷۳۷، ۷۳۸، ۷۳۹، ۷۴۰، ۷۴۱، ۷۴۲، ۷۴۳، ۷۴۴، ۷۴۵، ۷۴۶، ۷۴۷، ۷۴۸، ۷۴۹، ۷۵۰، ۷۵۱، ۷۵۲، ۷۵۳، ۷۵۴، ۷۵۵، ۷۵۶، ۷۵۷، ۷۵۸، ۷۵۹، ۷۶۰، ۷۶۱، ۷۶۲، ۷۶۳، ۷۶۴، ۷۶۵، ۷۶۶، ۷۶۷، ۷۶۸، ۷۶۹، ۷۷۰، ۷۷۱، ۷۷۲، ۷۷۳، ۷۷۴، ۷۷۵، ۷۷۶، ۷۷۷، ۷۷۸، ۷۷۹، ۷۸۰، ۷۸۱، ۷۸۲، ۷۸۳، ۷۸۴، ۷۸۵، ۷۸۶، ۷۸۷، ۷۸۸، ۷۸۹، ۷۹۰، ۷۹۱، ۷۹۲، ۷۹۳، ۷۹۴، ۷۹۵، ۷۹۶، ۷۹۷، ۷۹۸، ۷۹۹، ۸۰۰، ۸۰۱، ۸۰۲، ۸۰۳، ۸۰۴، ۸۰۵، ۸۰۶، ۸۰۷، ۸۰۸، ۸۰۹، ۸۱۰، ۸۱۱، ۸۱۲، ۸۱۳، ۸۱۴، ۸۱۵، ۸۱۶، ۸۱۷، ۸۱۸، ۸۱۹، ۸۲۰، ۸۲۱، ۸۲۲، ۸۲۳، ۸۲۴، ۸۲۵، ۸۲۶، ۸۲۷، ۸۲۸، ۸۲۹، ۸۳۰، ۸۳۱، ۸۳۲، ۸۳۳، ۸۳۴، ۸۳۵، ۸۳۶، ۸۳۷، ۸۳۸، ۸۳۹، ۸۴۰، ۸۴۱، ۸۴۲، ۸۴۳، ۸۴۴، ۸۴۵، ۸۴۶، ۸۴۷، ۸۴۸، ۸۴۹، ۸۵۰، ۸۵۱، ۸۵۲، ۸۵۳، ۸۵۴، ۸۵۵، ۸۵۶، ۸۵۷، ۸۵۸، ۸۵۹، ۸۶۰، ۸۶۱، ۸۶۲، ۸۶۳، ۸۶۴، ۸۶۵، ۸۶۶، ۸۶۷، ۸۶۸، ۸۶۹، ۸۷۰، ۸۷۱، ۸۷۲، ۸۷۳، ۸۷۴، ۸۷۵، ۸۷۶، ۸۷۷، ۸۷۸، ۸۷۹، ۸۸۰، ۸۸۱، ۸۸۲، ۸۸۳، ۸۸۴، ۸۸۵، ۸۸۶، ۸۸۷، ۸۸۸، ۸۸۹، ۸۹۰، ۸۹۱، ۸۹۲، ۸۹۳، ۸۹۴، ۸۹۵، ۸۹۶، ۸۹۷، ۸۹۸، ۸۹۹، ۹۰۰، ۹۰۱، ۹۰۲، ۹۰۳، ۹۰۴، ۹۰۵، ۹۰۶، ۹۰۷، ۹۰۸، ۹۰۹، ۹۱۰، ۹۱۱، ۹۱۲، ۹۱۳، ۹۱۴، ۹۱۵، ۹۱۶، ۹۱۷، ۹۱۸، ۹۱۹، ۹۲۰، ۹۲۱، ۹۲۲، ۹۲۳، ۹۲۴، ۹۲۵، ۹۲۶، ۹۲۷، ۹۲۸، ۹۲۹، ۹۳۰، ۹۳۱، ۹۳۲، ۹۳۳، ۹۳۴، ۹۳۵، ۹۳۶، ۹۳۷، ۹۳۸، ۹۳۹، ۹۴۰، ۹۴۱، ۹۴۲، ۹۴۳، ۹۴۴، ۹۴۵، ۹۴۶، ۹۴۷، ۹۴۸، ۹۴۹، ۹۵۰، ۹۵۱، ۹۵۲، ۹۵۳، ۹۵۴، ۹۵۵، ۹۵۶، ۹۵۷، ۹۵۸، ۹۵۹، ۹۶۰، ۹۶۱، ۹۶۲، ۹۶۳، ۹۶۴، ۹۶۵، ۹۶۶، ۹۶۷، ۹۶۸، ۹۶۹، ۹۷۰، ۹۷۱، ۹۷۲، ۹۷۳، ۹۷۴، ۹۷۵، ۹۷۶، ۹۷۷، ۹۷۸، ۹۷۹، ۹۸۰، ۹۸۱، ۹۸۲، ۹۸۳، ۹۸۴، ۹۸۵، ۹۸۶، ۹۸۷، ۹۸۸، ۹۸۹، ۹۹۰، ۹۹۱، ۹۹۲، ۹۹۳، ۹۹۴، ۹۹۵، ۹۹۶، ۹۹۷، ۹۹۸، ۹۹۹، ۱۰۰۰، ۱۰۰۱، ۱۰۰۲، ۱۰۰۳، ۱۰۰۴، ۱۰۰۵، ۱۰۰۶، ۱۰۰۷، ۱۰۰۸، ۱۰۰۹، ۱۰۱۰، ۱۰۱۱، ۱۰۱۲، ۱۰۱۳، ۱۰۱۴، ۱۰۱۵، ۱۰۱۶، ۱۰۱۷، ۱۰۱۸، ۱۰۱۹، ۱۰۲۰، ۱۰۲۱، ۱۰۲۲، ۱۰۲۳، ۱۰۲۴، ۱۰۲۵، ۱۰۲۶، ۱۰۲۷، ۱۰۲۸، ۱۰۲۹، ۱۰۳۰، ۱۰۳۱، ۱۰۳۲، ۱۰۳۳، ۱۰۳۴، ۱۰۳۵، ۱۰۳۶، ۱۰۳۷، ۱۰۳۸، ۱۰۳۹، ۱۰۴۰، ۱۰۴۱، ۱۰۴۲، ۱۰۴۳، ۱۰۴۴، ۱۰۴۵، ۱۰۴۶، ۱۰۴۷، ۱۰۴۸، ۱۰۴۹، ۱۰۵۰، ۱۰۵۱، ۱۰۵۲، ۱۰۵۳، ۱۰۵۴، ۱۰۵۵، ۱۰۵۶، ۱۰۵۷، ۱۰۵۸، ۱۰۵۹، ۱۰۶۰، ۱۰۶۱، ۱۰۶۲، ۱۰۶۳، ۱۰۶۴، ۱۰۶۵، ۱۰۶۶، ۱۰۶۷، ۱۰۶۸، ۱۰۶۹، ۱۰۷۰، ۱۰۷۱، ۱۰۷۲، ۱۰۷۳، ۱۰۷۴، ۱۰۷۵، ۱۰۷۶، ۱۰۷۷، ۱۰۷۸، ۱۰۷۹، ۱۰۸۰، ۱۰۸۱، ۱۰۸۲، ۱۰۸۳، ۱۰۸۴، ۱۰۸۵، ۱۰۸۶، ۱۰۸۷، ۱۰۸۸، ۱۰۸۹، ۱۰۹۰، ۱۰۹۱، ۱۰۹۲، ۱۰۹۳، ۱۰۹۴، ۱۰۹۵، ۱۰۹۶، ۱۰۹۷، ۱۰۹۸، ۱۰۹۹، ۱۱۰۰، ۱۱۰۱، ۱۱۰۲، ۱۱۰۳، ۱۱۰۴، ۱۱۰۵، ۱۱۰۶، ۱۱۰۷، ۱۱۰۸، ۱۱۰۹، ۱۱۱۰، ۱۱۱۱، ۱۱۱۲، ۱۱۱۳، ۱۱۱۴، ۱۱۱۵، ۱۱۱۶، ۱۱۱۷، ۱۱۱۸، ۱۱۱۹، ۱۱۲۰، ۱۱۲۱، ۱۱۲۲، ۱۱۲۳، ۱۱۲۴، ۱۱۲۵، ۱۱۲۶، ۱۱۲۷، ۱۱۲۸، ۱۱۲۹، ۱۱۳۰، ۱۱۳۱، ۱۱۳۲، ۱۱۳۳، ۱۱۳۴، ۱۱۳۵، ۱۱۳۶، ۱۱۳۷، ۱۱۳۸، ۱۱۳۹، ۱۱۴۰، ۱۱۴۱، ۱۱۴۲، ۱۱۴۳، ۱۱۴۴، ۱۱۴۵، ۱۱۴۶، ۱۱۴۷، ۱۱۴۸، ۱۱۴۹، ۱۱۵۰، ۱۱۵۱، ۱۱۵۲، ۱۱۵۳، ۱۱۵۴، ۱۱۵۵، ۱۱۵۶، ۱۱۵۷، ۱۱۵۸، ۱۱۵۹، ۱۱۶۰، ۱۱۶۱، ۱۱۶۲، ۱۱۶۳، ۱۱۶۴، ۱۱۶۵، ۱۱۶۶، ۱۱۶۷، ۱۱۶۸، ۱۱۶۹، ۱۱۷۰، ۱۱۷۱، ۱۱۷۲، ۱۱۷۳، ۱۱۷۴، ۱۱۷۵، ۱۱۷۶، ۱۱۷۷، ۱۱۷۸، ۱۱۷۹، ۱۱۸۰، ۱۱۸۱، ۱۱۸۲، ۱۱۸۳، ۱۱۸۴، ۱۱۸۵، ۱۱۸۶، ۱۱۸۷، ۱۱۸۸، ۱۱۸۹، ۱۱۹۰، ۱۱۹۱، ۱۱۹۲، ۱۱۹۳، ۱۱۹۴، ۱۱۹۵، ۱۱۹۶، ۱۱۹۷، ۱۱۹۸، ۱۱۹۹، ۱۲۰۰، ۱۲۰۱، ۱۲۰۲، ۱۲۰۳، ۱۲۰۴، ۱۲۰۵، ۱۲۰۶، ۱۲۰۷، ۱۲۰۸، ۱۲۰۹، ۱۲۱۰، ۱۲۱۱، ۱۲۱۲، ۱۲۱۳، ۱۲۱۴، ۱۲۱۵، ۱۲۱۶، ۱۲۱۷، ۱۲۱۸، ۱۲۱۹، ۱۲۲۰، ۱۲۲۱، ۱۲۲۲، ۱۲۲۳، ۱۲۲۴، ۱۲۲۵، ۱۲۲۶، ۱۲۲۷، ۱۲۲۸، ۱۲۲۹، ۱۲۳۰، ۱۲۳۱، ۱۲۳۲، ۱۲۳۳، ۱۲۳۴، ۱۲۳۵، ۱۲۳۶، ۱۲۳۷، ۱۲۳۸، ۱۲۳۹، ۱۲۴۰، ۱۲۴۱، ۱۲۴۲، ۱۲۴۳، ۱۲۴۴، ۱۲۴۵، ۱۲۴۶، ۱۲۴۷، ۱۲۴۸، ۱۲۴۹، ۱۲۵۰، ۱۲۵۱، ۱۲۵۲، ۱۲۵۳، ۱۲۵۴، ۱۲۵۵، ۱۲۵۶، ۱۲۵۷، ۱۲۵۸، ۱۲۵۹، ۱۲۶۰، ۱۲۶۱، ۱۲۶۲، ۱۲۶۳، ۱۲۶۴، ۱۲۶۵، ۱۲۶۶، ۱۲۶۷، ۱۲۶۸، ۱۲۶۹، ۱۲۷۰، ۱۲۷۱، ۱۲۷۲، ۱۲۷۳، ۱۲۷۴، ۱۲۷۵، ۱۲۷۶، ۱۲۷۷، ۱۲۷۸، ۱۲۷۹، ۱۲۸۰، ۱۲۸۱، ۱۲۸۲، ۱۲۸۳، ۱۲۸۴، ۱۲۸۵، ۱۲۸۶، ۱۲۸۷، ۱۲۸۸، ۱۲۸۹، ۱۲۹۰، ۱۲۹۱، ۱۲۹۲، ۱۲۹۳، ۱۲۹۴، ۱۲۹۵، ۱۲۹۶، ۱۲۹۷، ۱۲۹۸، ۱۲۹۹، ۱۳۰۰، ۱۳۰۱، ۱۳۰۲، ۱۳۰۳، ۱۳۰۴، ۱۳۰۵، ۱۳۰۶، ۱۳۰۷، ۱۳۰۸، ۱۳۰۹، ۱۳۱۰، ۱۳۱۱، ۱۳۱۲، ۱۳۱۳، ۱۳۱۴، ۱۳۱۵، ۱۳۱۶، ۱۳۱۷، ۱۳۱۸، ۱۳۱۹، ۱۳۲۰، ۱۳۲۱، ۱۳۲۲، ۱۳۲۳، ۱۳۲۴، ۱۳۲۵، ۱۳۲۶، ۱۳۲۷، ۱۳۲۸، ۱۳۲۹، ۱۳۳۰، ۱۳۳۱، ۱۳۳۲، ۱۳۳۳، ۱۳۳۴، ۱۳۳۵، ۱۳۳۶، ۱۳۳۷، ۱۳۳۸، ۱۳۳۹، ۱۳۴۰، ۱۳۴۱، ۱۳۴۲، ۱۳۴۳، ۱۳۴۴، ۱۳۴۵، ۱۳۴۶، ۱۳۴۷، ۱۳۴۸، ۱۳۴۹، ۱۳۵۰، ۱۳۵۱، ۱۳۵۲، ۱۳۵۳، ۱۳۵۴، ۱۳۵۵، ۱۳۵۶، ۱۳۵۷، ۱۳۵۸، ۱۳۵۹، ۱۳۶۰، ۱۳۶۱، ۱۳۶۲، ۱۳۶۳، ۱۳۶۴، ۱۳۶۵، ۱۳۶۶، ۱۳۶۷، ۱۳۶۸، ۱۳۶۹، ۱۳۷۰، ۱۳۷

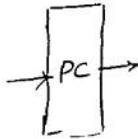


③



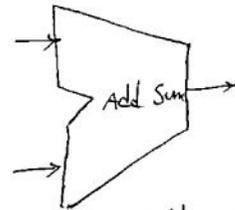
a) instruction memory

۱. واحد حافظه برای ذخیره سازی دستورالعمل ها در نام نیاز داریم  
سین کلا هر دستورالعمل آدرس را تکرار می کنیم



b) program counter

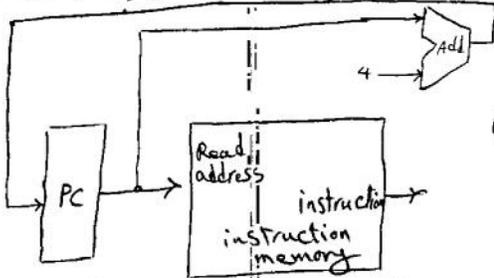
۲. یک شمارنده برای Program Counter بر روی PC  
ثبت Register آدرس دستورالعمل می  
رابط می طرد.



c) Adder

۳. جمع کننده. به ترتیب کار مشخص کردن آدرس دستورالعمل کلا هر یک در PC به نام  
این جمع کننده مداری تماماً ترکیبی است.

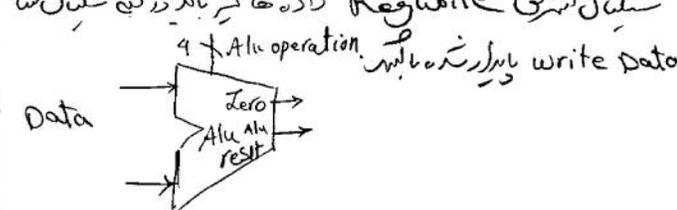
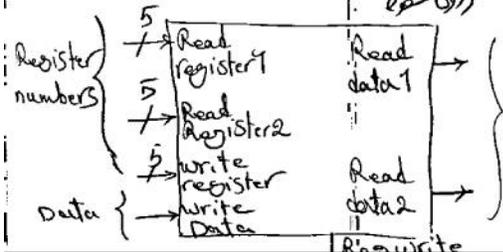
توضیح: بجای جای یک دستورالعمل در محاسبه هم باید آن دستورالعمل را از حافظه واقعی fetch کنیم همین برای  
اجرای دستورالعمل بعدی باید مقدار کنونی PC را با انداز ۴ واحد افزایش داد تا به دستورالعمل ثبت سر اشاره کند.

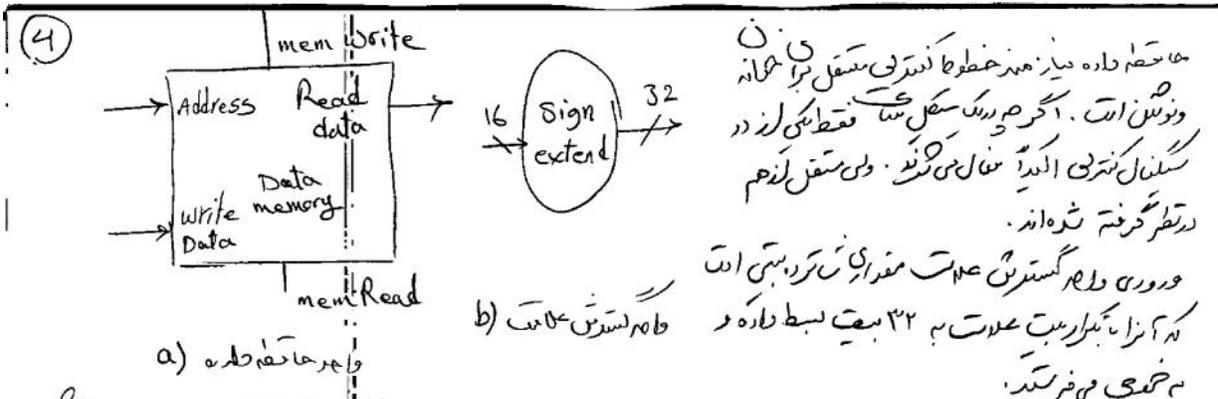


دستورالعمل ها نوع R: ثابت دستورالعمل های این رده در بیت  
را محاسبه و یکی از عملیات ALU (محاسبه با منطقی) را بر روی آنها  
انجام می دهد و در نهایت نتیجه را در بیت های دیگر می نویسد. در راه  
حساب add - and - or - Bus و SLT است.

add st1, st2, st3 ← برای اجرا باید بیت st2 و st3 را خواند و نتیجه را در بیت st1 نوشت.  
در مدله مورد نیاز برای پیاده سازی عملیات محاسباتی منطقی از نوع بیت R-type عبارتند از: باقی بیت و یک ALU  
باقی بیت متضمن به تمام بیت ها در درجه (مورد) کار خواندن و یک درجه کار نوشتن است.

باقی بیت: همیشه محتوای بیت هایی را در خروجی های خود در خط می سازد که شماره آن بیت ها در دو خط Read/Ready و در دو خط  
مستحق شده است. هیچ سگنل کنترلی دیگری برای عملیات نیاز نیست. برای نوشتن درون یکی از بیت ها حسابی  
بیت را در دو خط write مستحق شده باشد و اقرون در فعال کردن  
سگنل کنترلی Register داده ها نیز باید در یکی از سگنل های Read/Ready خط



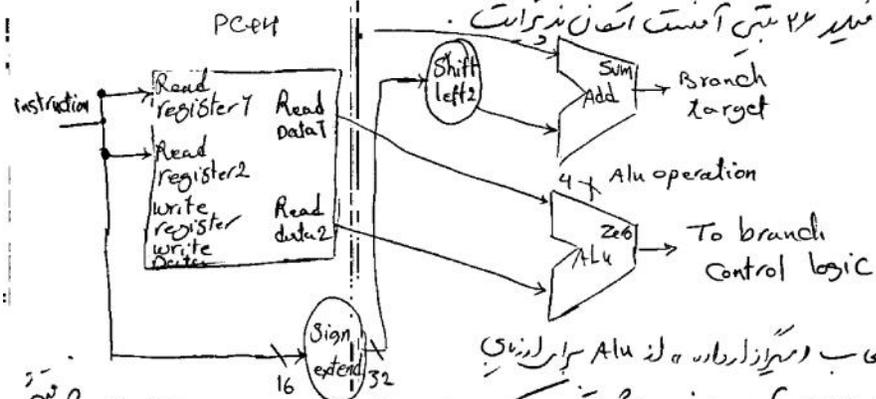


← برای پایداری این دست‌نویس عملیات آدرس‌دهی از گسترش عملیات مقدار PC استفاده می‌شود تا آدرس هدف همیشه درست آید.

۱- می‌تواند در هدف آدرس‌دهی با گسترش عملیات وین عملیات خواهد بود

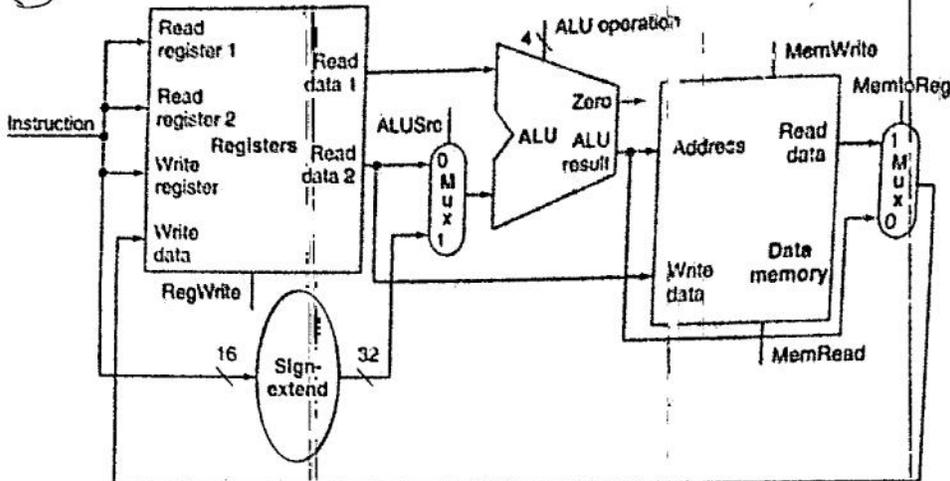
۲- مقادیر گم‌شده‌ها به بارهای آدرس‌دهی نسبت به گسترش دو عملیات در نظر گرفته می‌شوند.

دست‌نویس غیر شرطی (ن) درین نحو عمل می‌کند. ۲۸ بیت کم‌ارزش‌ترین PC را ۲۶ بیت از آدرس درج شده درین دست‌نویس، پس از دو بیت نسبت به چپ، جایگزین می‌کند. در حقیقت نسبت به چپ به اندازه دو بیت به چپ آدرس در دو بیت صفر است راست از مقدار ۲۶ بیتی آدرس نشان می‌دهد.



توضیح: برای اجرای دست‌نویس انتخاب و می‌تواند از ALU برار آدرس شرطی و از جمع‌کننده اجرا می‌کند. برای آدرس باید ۴ بیت جمع مقدار آدرس با PC (یعنی PC44) مقدار آدرس گسترش عملیات و نسبت به اندازه دو بیت به چپ آدرس داده می‌شود. Shift left 2 شیفت داده شده با اضافه کردن ۰۰۰ به سمت راست عدد مقدار در درج‌ها را می‌تواند به شخصی نشان دهد. مدارات کنترل اساسی شخصی سفید Zero از ALU تصمیم می‌گیرد که آیا آدرس در دست‌نویس PC44 و آدرس می‌تواند، هدف می‌تواند که باید درون PC جایگزین شود.

5



شکل:

مسیر گذر داده برابر دستور العمل در چی لایه منطقی. همچنین از نوع R در متن بر بیت هستند از دستور العمل در رجوع به حافظه نسبتاً نسبی است. همچنین دستورالعمل مسیری گذر داده برابر بخش عملیاتی دستور العمل در رجوع به حافظه همچنین دستورالعمل های عملیاتی منطقی است به قسمی که دستورالعملی از یک باغیاتی است Register File و یک ALU و در برابر اجرا هر دو در این دستورالعمل ها همه حرفت هر کجا لازم باشد از حالتی کلیتر به کلیتر در.

مجاب : به اینجا دین می آید که در داده که بر روی آن تنها یک باغیاتی است و یک ALU وجود داشته باشد در ورودی ALU از دو دسته ورودی که منبع متفاوتی دارند نسبتیاتی کنیم. همچنین ترتیب برابر خطوط ورودی باغیاتی است (که به منظور ترتیب در دست به تعبیر شده) باید از دسته ورودی که از برای متفاوتی است نسبتیاتی شود. بدین سان باغیاتی کلیتر در ورودی ALU و یک حالتی کلیتر در ورودی باغیاتی است که در داده شود.

Address 15:0

برای واحد کنترل مرکزی:

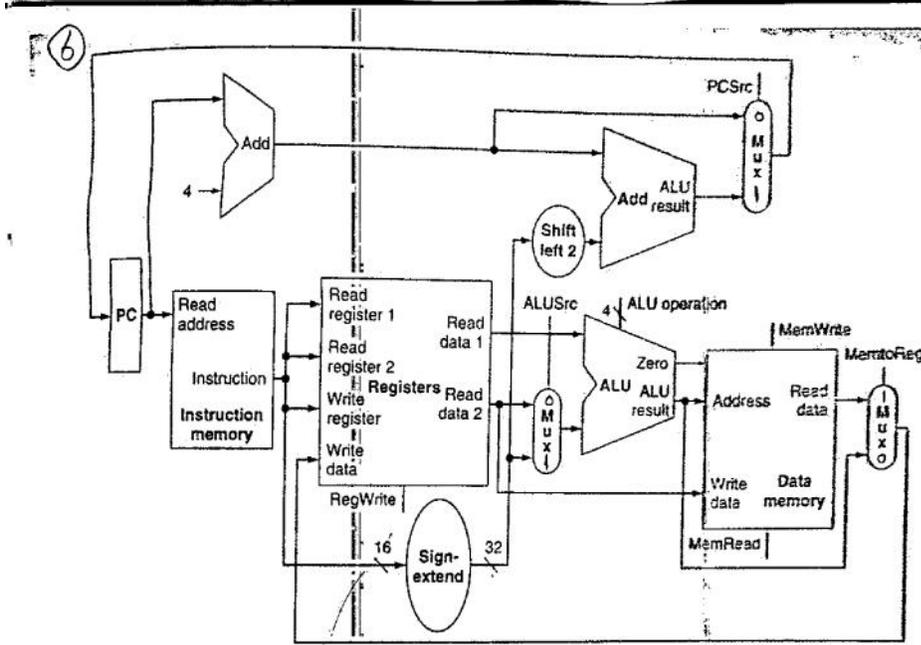
0	rs	rt	rd	shamt	funct
31:26	25:21	20:16	15:11	10:6	5:0

به فیلد دوم به opcode که در آن که عمل درج هر جز همیشه در بوقیت بیت های 26 تا 31 قرار دارد به این فیلد با را

op[5:0] است. خواهیم کرد

\* درستی آن با بستن شماره خواننده همیشه در فیلدها دوم به 25 و 26 مشخص می شوند که به ترتیب در بوقیت بیت های 21 و 25 و 16 تا 20 قرار دارند. این موضوع تکرار نام دستورالعمل از نوع R سرش شروع است و در بعضی دستورالعمل زخمه در حافظه من میگذرد. هدف از تکرار بیت آنست که در بعضی دستورالعمل هر سرش شروع است و در بعضی دستورالعمل از خیمه در بارگذاری وجود دارد در بوقیت بیت های 15 تا 20 قرار می گیرند

\* در دستوراتی که در بعضی از خیمه در بارگذاری همیشه در بوقیت بیت های 21 تا 25 میروم به فیلد که قرار دارند نسبت مقصد یکی از در بوقیت مشخص خواهد شد. در دستورالعمل بارگذاری Load در بوقیت بیت های 16 تا 20 فیلد دوم که قرار دارد در حالتی که دستورالعمل از نوع R یعنی دستورالعمل بارشروع است همین فیلد در بوقیت بیت های 11 تا 15 در فیلد 1 واقع است. به همین دلیل برای مشخص کردن اندازه بیت مقصد که باید در آن آن بزرگ تر در حالتی کلیتر به کلیتر در.

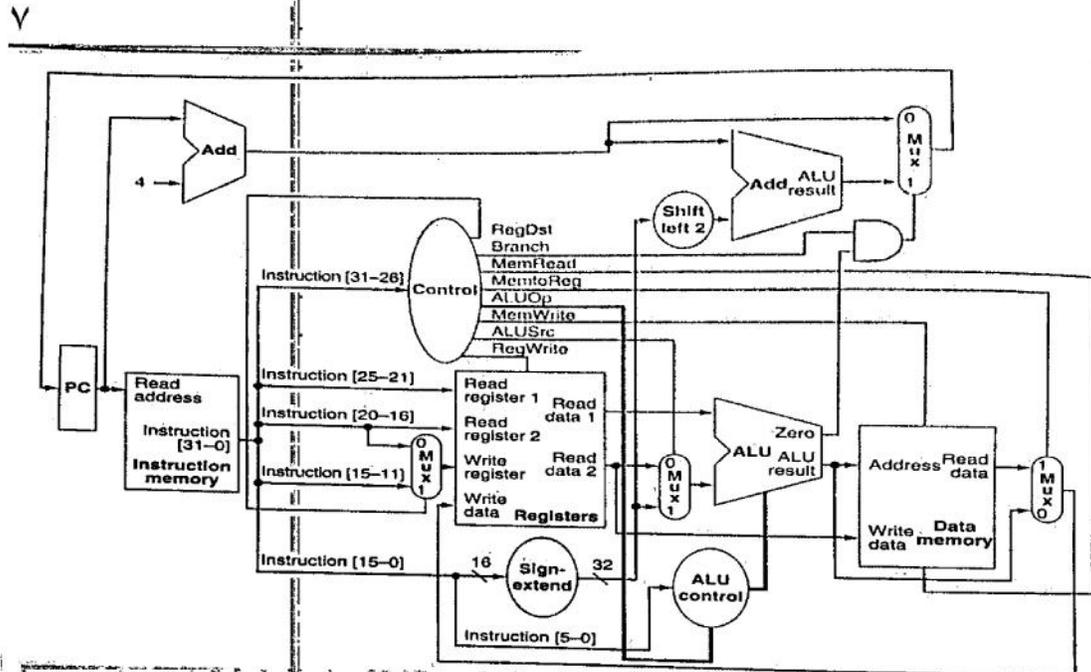


PCSrc: این خط کنترل در شرط فعال می‌شود. اول آنکه دستور العمل در حال اجرا «تغییر می‌شود» به معنی « $PC + 4$ »  
 (تصمیمی که داده می‌شود می‌تواند بر اساس فیلد opcode اتخاذ کند)  
 دوم آنکه خروجی Zero از ALU (که بر اساس نتیجه عملیات در آنجا می‌شود) مولد فعال باشد  
 نه بر این اساس که می‌تواند سیگنال PCSrc صادر خواهد داشت تا سیگنال دیگری از داده‌ها که می‌تواند سیگنال Branch  
 و سیگنال Zero از خروجی ALU را با هم ترکیب کند.

(در صفحه ۷)

مثال ۲: دستور العمل  $ST1, offset, ST2$  را شرح دهید.

- ۱- دستور العمل مورد نظر را می‌توان Fetch شده و شماره برنامه PC افزایش می‌دهد.
- ۲- مقدار بیت (در این مثال ST2) از روی مایعای بیت خوانده می‌شود.
- ۳- ALU حاصل جمع مقدار که از مایعای بیت خوانده شده و نت گرفته بیت کم (منفی دستور العمل یعنی مقدار offset) را پس از گسترش علامت Sign-extend حاصل می‌شود.
- ۴- حاصل جمع بیت آمده از ALU میزان آدرس داده در حافظه مورد اشاره قرار می‌دهد.
- ۵- داده‌ای که داده می‌شود به دست می‌آید درون یکی از بیت‌ها مایعای بیت ترکیب خواهد کرد. بیت مقدار در بیت ۱۶ تا ۲۵ از دستور العمل مشخص شده است. (در این مثال ST1)

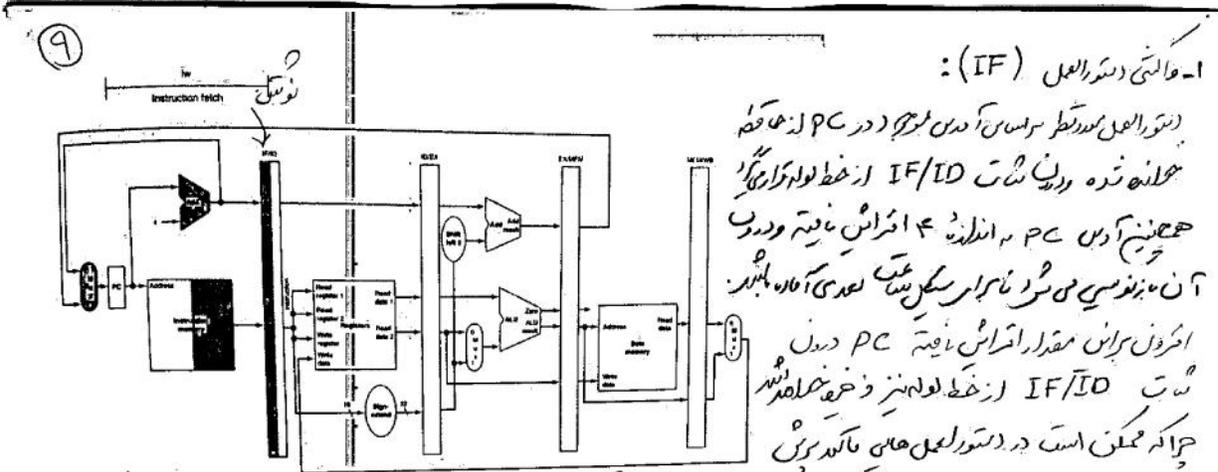


**شکل ۴-۱۷** «مسیر گذار داده» به همراه واحد کنترل، ورودی واحد کنترل، فیلد شش بیتی opcode از بطن کد دستورالعمل است. خروجی‌های واحد کنترل عبارتند از ۳ سیگنال کنترلی که مالتی‌پلکسرها را کنترل می‌کنند (سیگنال‌های  $ALUSrc$ ،  $RegDst$  و  $MemtoReg$ )، سه تای دیگر نیز برای کنترل و راهبری عملیات خواندن و نوشتن از «جایگانی ثبت» ( $register\ file$ ) و «حافظه» تولید می‌شود (سیگنال‌های  $MemWrite$  و  $MemRead$ ،  $RegWrite$ ،  $ALUOp$ ،  $ALUSrc$ ،  $MemWrite$  و  $MemRead$ ). سیگنال تک‌بیتی  $Branch$  مشخص می‌کند که آیا یک دستورالعمل  $AND$  انشعاب در حال اجراست؟ دو بیت سیگنال کنترلی  $ALUOp$  برای کنترل  $ALU$  مورد استفاده قرار خواهد گرفت. یک گیت  $AND$  برای کنترل مقدار بعدی  $PC$  سیگنال تولید می‌کند؛ اگر دستورالعمل از نوع انشعاب باشد و مقادیر عملوندها مساوی بودند مقدار بعدی  $PC$  از طریق مالتی‌پلکسر با «آدرس هدف» پریشن» جایگزین می‌شود.

مثال: عملکرد مسیر پردازنده  $datapath$  برای دستورالعمل نوع  $R$  همانند  $add\ \$t1,\ \$t2,\ \$t3$  همانند زیر است.  
 همه محدودیت در یک سطح است اما قیاسی می‌توان در اجرای دستورالعمل هر مرحله را مقدر کنیم.

- ۱- دستورالعمل و آنتی شده مقدار شمارنده برنامه  $PC$  افزایش خواهد یافت
- ۲- در بیت  $\$t2$  و  $\$t3$  از درون مالتی‌پلکس  $register\ file$  خوانده می‌شود. افزودن بر این و این کنترل در محدود این مرحله تنظیمات خطوط کنترل را می‌توانم خواهد کرد.
- ۳-  $ALU$  بر روی داده در خوانده شده از مالتی‌پلکس و بر اساس کد عمل (یعنی فیلدش  $Func$  در بیت‌ها) عمل کند (دستورالعمل)، عمل مطلب را انجام داده و نتیجه را در خروجی تولید می‌کند.
- ۴- بر اساس بیت‌های ۱۱ تا ۱۵ از دستورالعمل که نگه داشته مقدار تعیین می‌کند. در این مثال  $\$t1$  نتیجه خروجی از  $ALU$  در مالتی‌پلکس ثبت نوشته می‌شود.
- مثال ۳:  $beq\ \$t1,\ \$t2,\ offset$  ← ۱- دستورالعمل و آنتی  $Fetch$  شده و در دستورالعمل  $PC$  افزایش می‌یابد.
- ۲- در بیت  $\$t1$  و  $\$t2$  از درون مالتی‌پلکس  $register\ file$  خوانده می‌شوند.
- ۳-  $ALU$  مقدار  $offset$  را از درون مالتی‌پلکس  $PC$  و  $PC+4$  با مقدار  $PC+4$  با مقدار  $PC+4$  از دستورالعمل یعنی مقدار  $offset$  بر از کنترل عملیات و تنظیم  $PC$  در این مرحله خواهد کرد. در این مرحله  $PC$  در دستورالعمل خواهد بود.
- ۴- از نتیجه  $ALU$  در مالتی‌پلکس  $Zero$  که  $ALU$  بر اساس مقایسه  $\$t1$  و  $\$t2$  در این مرحله  $PC$  در دستورالعمل خواهد بود.

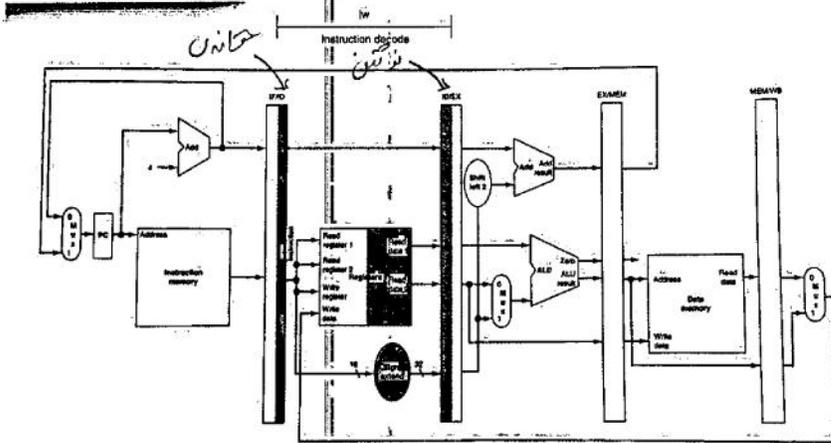




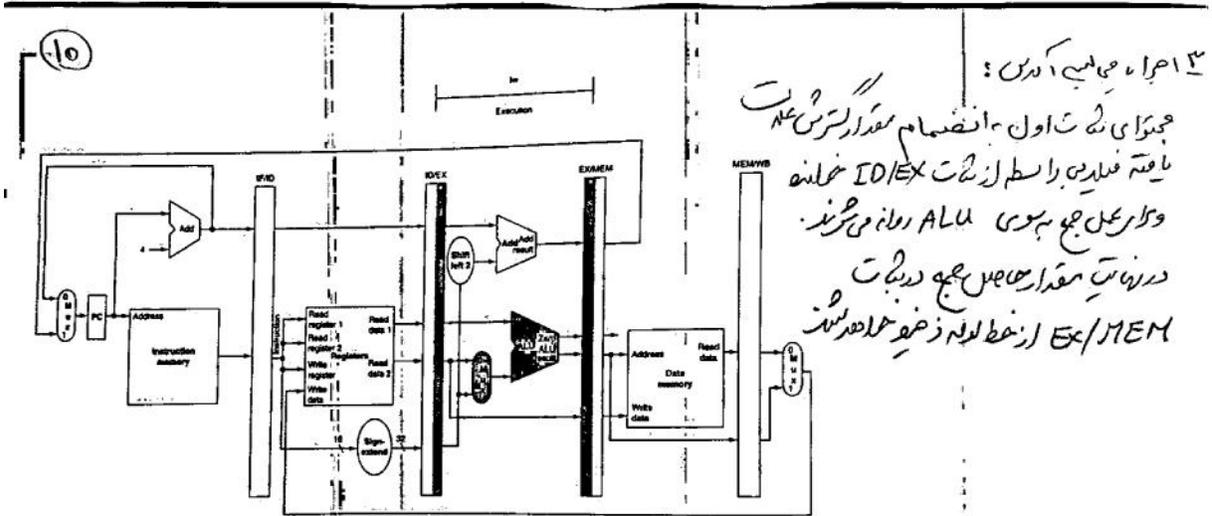
۱- حالتی دستورالعمل (IF):

دستورالعمل مورد نظر بر اساس آدرس موجود در PC از حافظه خوانده شده و در این حالت IF/ID از خط لوله قرار می‌گیرد. همچنین آدرس PC به اندازه ۴ افزایش یافته و در آن به بزرگترین می‌شود تا امکان ساخت بعدی آدرس باشد. اکنون برای مقدار آفرایش یافته PC درون حالت IF/ID از خط لوله نیز ذخیره می‌شود چرا که ممکن است در دستورالعمل‌های فاکتور

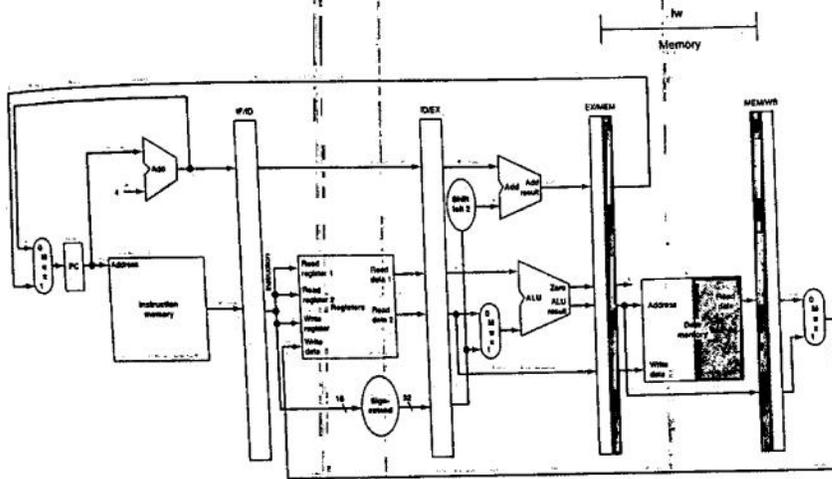
شرطی beq در عملیات می‌تواند به خط لوله به مقدار آن نیاز داشته باشیم. کامپیوتر در این مرحله هنوز قادر به تشخیص «درخت» دستورالعمل و آفرایش شده نیست. فلذا برای آماده سازی دستورالعمل خوانده شده برای عملیات گذشتی decode، مقدار دستورالعمل به همراه اطلاعاتی که ممکن است در مرحله بعد مورد نیاز باشد از طریق بگت IF/ID به مرحله دوم از خط لوله روانه می‌گردد.



۲- گذشتگی دستورالعمل و خواندن از بگت‌های : فیلدین وابسته immediate Field از بخش‌های دستورالعمل از بگت IF/ID به سمت شارژر بلتس استخراج شده و به یک مقدار ۳۲ بیتی گسترش می‌دهد Sign-extension. داده‌ها در بخش‌های دیگر دستورالعمل به عنوان شماره بگت‌هایی که باید عملیات استخراج و به بگت‌های بگت‌ها استفاده شود. این به مقدار چهار آدرس افزایش یافته PC در بگت ID/EX از خط لوله ذخیره می‌شود. ما در بخش‌های بعدی خواهیم دید که چگونه است در سطح‌های بعدی برای این دستورالعمل مورد نیاز باشد از طریق بگت ID/EX به مرحله بعد انتقال می‌دهیم.

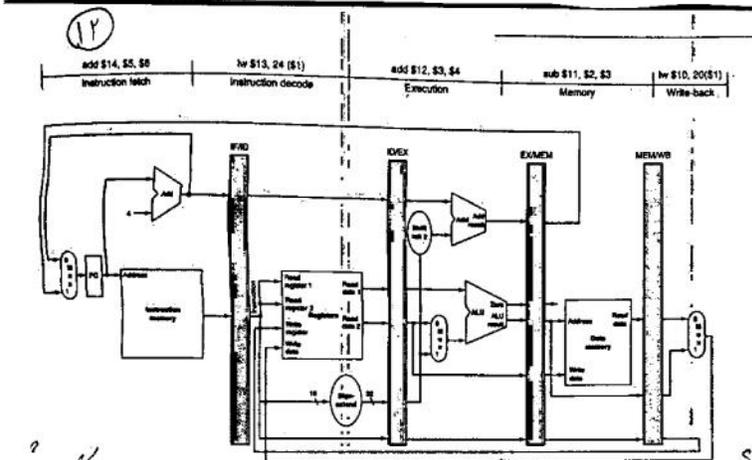


توجه: همواره در دارن هر چیزی که در مرحله ابتدایی به هر حال استی می تواند، اما این با این درونی که خط لوله قرار دارد، سرد و سرد  
 به محض آنکه دستور العمل بدست می آید مرحله ای از خط لوله سرد اطلاعات جدیدی آن مرحله از دست خواهد رفت.



ع - مرحله جمع حافظه؛ با استفاده از کسری موجود در ثبت EX/MEM از خط لوله، مجموعی که از حافظه خوانده می شود، در ثبت  
 خوانده شده در ثبت MEM/WB از خط لوله ذخیره خواهد شد.





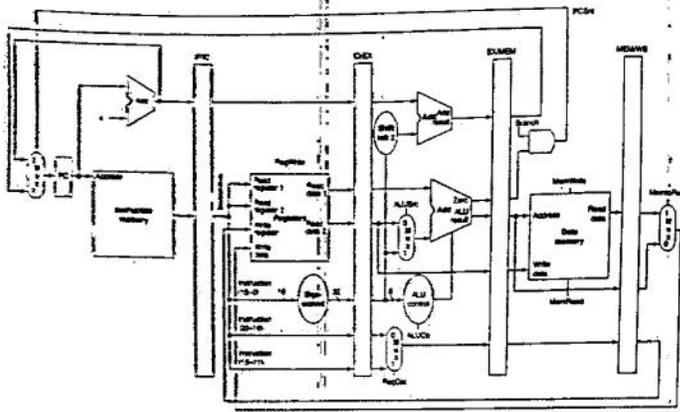
نمودار تک سیکل ساعتی  
Single-clock-cycle diagram

Lw \$t0, 20(\$t1)  
Sub \$t1, \$t2, \$t3  
add \$t2, \$t3, \$t4  
Lw \$t3, 24(\$t1)  
add \$t4, \$t5, \$t6

در این نمودار زمان از سمت چپ به سمت راست

صفحه به صفحه و در شرایطی که با هم میزنند یعنی در هر مرحله از اجرای روی می آید هر سه مرحله نام گذاری می شود

ع. خطوط انتزاعی مرتبط با دسترسی به حافظه؛ خط اول آنکه در این مرحله با بردار می رفته می شود عبارتند از MemRead، Branch و MemWrite این سگنالها بر حسب ترتیب در سوال اول در بر سر شرط بیت های beq، با برداری در زیر صفحه مقدار می دهند و تنظیم خواهند شد  
د. خط اول آنکه در شرط با برداری در سگنال انتزاعی برای این مرحله خط اول عبارتند از RegWrite، MemtoReg  
سگنال MemtoReg از بین دو مقدار خروجی ALU و مقدار خروجی از حافظه می گذارد که برای نوشتن در رجیستر است  
انتخاب می کند سگنال RegWrite مقدار انتخاب شده را در رجیستر می نویسد

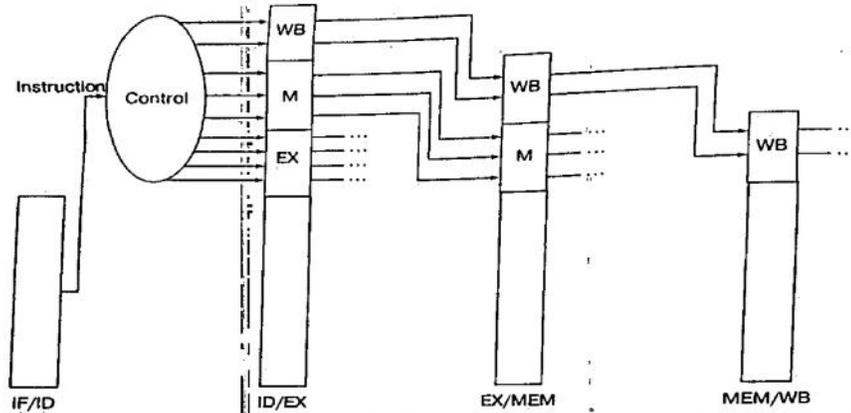


واحد کنترل خط اول؛  
مهره پردازنده دارد خط اول ایالت پیپلیند  
پردازنده سگنال انتزاعی

۱- خط اول آنکه در شرط با برداری در سوال اول؛  
از آنجایی که خط اول کنترل لازم برای خواندن شرط از حافظه  
در سوال اول مایکروسافت در PC همیشه نوآوری فعال  
است بنابراین هر چه در این مرحله خط اول در این

کنترل و کنترل داده با هم میزنند  
۲- خط اول آنکه در شرط با برداری در سوال اول با خواندن شرط از حافظه شرط را میخواند  
و خط انتزاعی خاص مایکروسافت و جود میگذارد که نشان می دهد مقدار در رجیستر است  
۳- خط اول آنکه در شرط با برداری در سوال اول؛ سگنال در این مرحله می باشد مقدار در رجیستر و تنظیم می شود عبارتند از: RegDst  
ALUSrc، ALUop این سگنالها بر حسب ترتیب در سوال اول در بر سر شرط بیت های beq، با برداری در زیر صفحه مقدار می دهند و تنظیم خواهند شد  
در سوال اول مایکروسافت در PC همیشه نوآوری فعال  
است بنابراین هر چه در این مرحله خط اول در این  
کنترل و کنترل داده با هم میزنند  
سگنال ALUSrc از بین دو مقدار خروجی ALU و مقدار خروجی از حافظه می گذارد که برای نوشتن در رجیستر است  
انتخاب می کند سگنال RegWrite مقدار انتخاب شده را در رجیستر می نویسد

۱۳



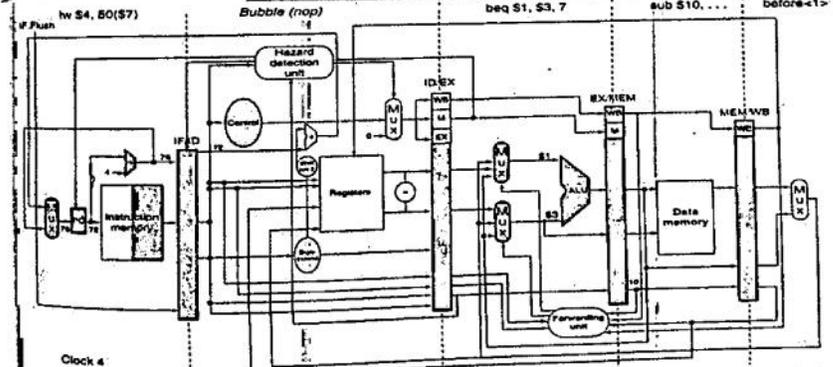
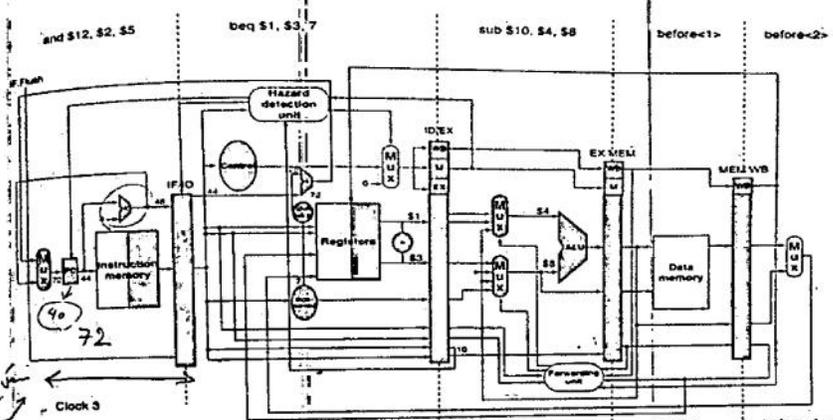
شکل ۴-۵ خطوط کنترلی برای سه مرحله آخر از خط لوله. دقت کنید که ۴ سیگنال از ۳ سیگنال کنترلی در مرحله EX مورد استفاده قرار می‌گیرند در حالی که ۵ سیگنال باقیمانده از طریق تبات EX/MEM به مراحل بعدی تحویل داده می‌شوند. از این ۵ سیگنال ۳ تای آنها در مرحله MEM بهره‌برداری شده و دو تای باقیمانده یکبار دیگر از طریق تبات MEM/WB به آخرین مرحله از خط لوله یعنی «مرحله بازنویسی» (WB Stage) تحویل خواهد شد.

شکل ۴-۶: نتایج اجرای برنامه (رابطه‌های ریاضی) همان‌طور که در بخش قبلی

پاسخ: شکل ۴-۶ تمام وقایعی را که در صورت انجام پرش اتفاق خواهد افتاد به تفصیل کشیده است. برخلاف نمودار شکل ۴-۶، در این شکل تنها یک «جاب» بر روی خط لوله ظاهر می‌شود.

```

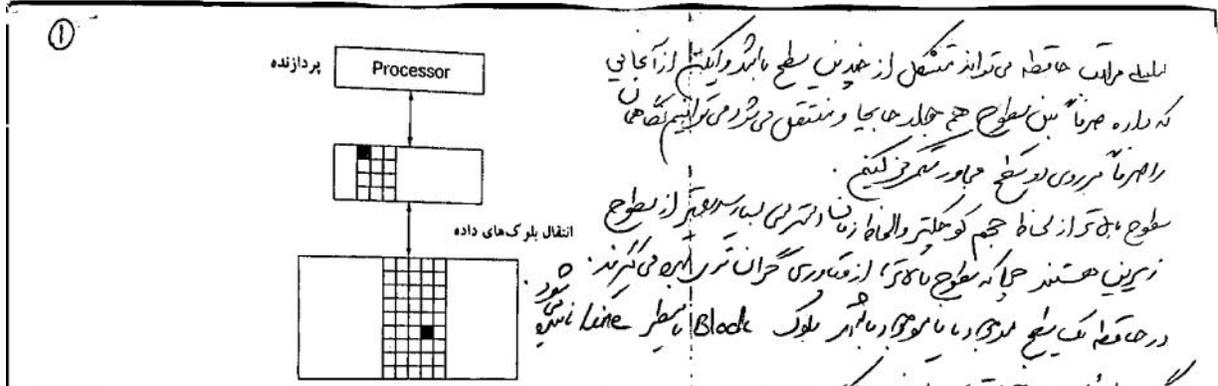
36 sub $t0, $t4, $t8
40 beq $t1, $t3, 7
44 and $t2, $t2, $t5
48 or $t3, $t2, $t6
52 add $t4, $t4, $t2
56 slt $t5, $t6, $t7
72 lw $t4, 50($t7)
    
```



شکل ۴-۷ در مرحله ۱۱۱ از سیکل ساعت سوم مشخص می‌شود که انجام پرش قطعی است و بنابراین با انتخاب ۷۲ به عنوان آدرس دستورالعمل بعدی، این مقدار در درون PC ذخیره شده و دستورالعمل در حال واکنش به صفر مقداردهی می‌شود.

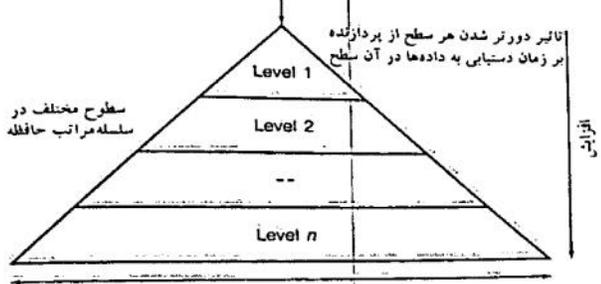
## فصل

## پنجم



شکل ۲-۵ می توان تصور کرد که هر جفت از سطوح سلسله مراتب حافظه دارای یک سطح مافوق و یک سطح مادون است و داده ها بین سطوح هم جوار منتقل می شوند. در هر سطح، کوچکترین واحد اطلاعات که یا در آن سطح حاضر و یا غایب است اصطلاحاً «بلوک» یا «سطح» نامیده می شود. معمولاً وقتی چیزی را بین دو سطح مجاور منتقل و کپی می کنیم یک بلوک کامل انتقال می یابد.

hit rate: کسر از کل مراجعات که در حافظه یافت می شود.  
 miss rate: کسر از کل مراجعات ناموفق که داده در حافظه یافت نمی شود و باید به سطح پایین تر مراجعه کند.  
 hit time: زمانی که صرف می شود تا داده در حافظه یافت شود.  
 miss time: زمانی که صرف می شود تا داده در حافظه یافت نشود و باید به سطح پایین تر مراجعه کند.

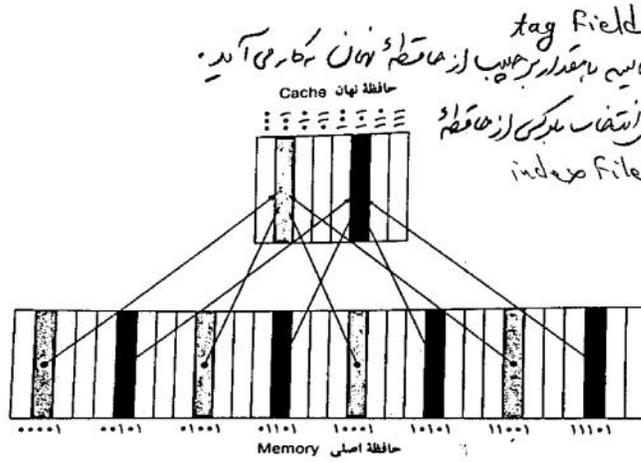


شکل ۳-۵ این نمودار ساختار یک حافظه سلسله مراتبی را به تصویر کشیده است؛ هر چه از پردازنده دورتر شویم ظرفیت آن افزایش خواهد یافت. این ساختار با فرض بهره گیری از مکانیزم های عملیاتی مناسب، این امکان را فراهم می آورد تا زمان دسترسی پردازنده به حافظه در حدود زمان دسترسی به داده ها در سطح نخست باشد ولی گامکان ظرفیتی مبادله با آخرین سطح (سطح نام) را در اختیار بگذارد. حفظ چنین پنداشتی موضوع مباحث این فصل است. اگرچه دیسک های مغناطیسی بطور نامتداول در پایین ترین سطح از سلسله مراتب حافظه قرار می گیرند ولیکن در برخی از سیستم ها در پایین ترین سطح، از نوارهای مغناطیسی یا سرویس دهنده های قابل موجود بر روی شبکه محلی استفاده می کنند و آنها را نیز به عنوان سطح بعدی از سلسله مراتب حافظه به حساب می آورند.

direct mapped: ساده ترین روش برای تعیین واحدهای مشخص برای عملیات در حافظه است. این روش است که هر قطعه حافظه را به یک آدرس حقیقی خاص نگه می دارد و حافظه اصلی را به همان آدرس حافظه اختصاص می دهد. این روش ساده ترین و سریع ترین روش برای تعیین واحدهای مشخص برای عملیات در حافظه است. این روش است که هر قطعه حافظه را به یک آدرس حقیقی خاص نگه می دارد و حافظه اصلی را به همان آدرس حافظه اختصاص می دهد. این روش ساده ترین و سریع ترین روش برای تعیین واحدهای مشخص برای عملیات در حافظه است.

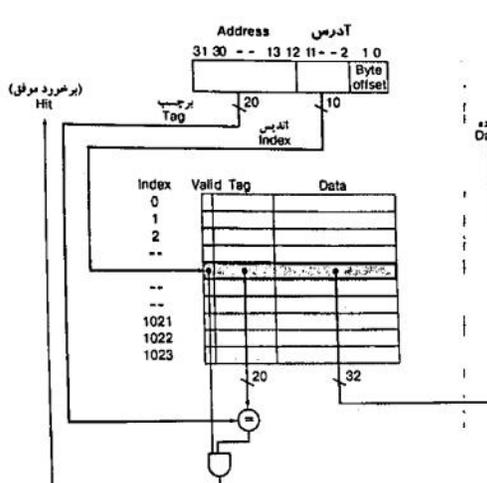
Block Address Mod (number of blocks in the cache)  
 Mod (تعداد بلوکها در حافظه) (آدرس بلوک)

دائری «تفاوت مستقیم» direct mapped ما با دست راستی یک آدرس تنها می توانیم یک بلاک از حافظه را در آن نگه داریم زیرا این بیت های کم آدرس و تنها شماره یک بلاک از حافظه زمان را تعیین می کند.



**شکل 5-5** یک حافظه نهان مبتنی بر «نگاشت مستقیم» با ظرفیت هشت درایه. این شکل نشان می دهد که چگونه کلمات واقع در آدرس ۰ تا ۳۱ از حافظه اصلی در موقعیت های مشابهی از حافظه نهان نگاشته می شوند. از آنجایی که ظرفیت حافظه نهان فقط هشت کلمه است بنابراین آدرس مفروض X مستقیماً به کلمه ای با آدرس  $X \bmod 8$  از حافظه نهان نگاشته خواهد شد. بنابراین سه بیت کم آدرس از آدرس هر کلمه در حافظه اصلی، محل قرارگیری آن کلمه در حافظه نهان را تعیین می کند. با این تفسیر تمام کلمات واقع در آدرس های ۰۰۰۰، ۰۱۰۰، ۱۰۰۰، ۱۱۰۰ همگی به موقعیت مشابهی از حافظه نهان نگاشته می شوند.

۴ آدرس های ۳۲ بیتی برای هر بیت حافظه  
 ۴ حافظه نهان «تفاوت مستقیم»  
 ۴ حافظه نهان متصل را  $2^n$  بلاک و طبعاً  $n$  بیت نیاز است.  
 \* جزل هر بلاک  $2^m$  کلمه چهارمادسی (معالی)  $2^{m+2}$   
 باقی است فلذا  $m$  بیت سرگرس در هر بلاک  
 بدون هر بلاک حافظه نهان رد و بیت سرگرس آدرس  
 در هر حافظه از یک کلمه نیاز داریم.  
 این اوصاف نهانی قدری عجیب و غریب معادل  
 $32 - (n + m + 2)$   
 بیت است. و در هر کلمه بیت های مورد  
 نیاز سرگرس حافظه نهان با شصت  
 بار عبارت است از:



**شکل 5-6** در این حافظه نهان بخش گنجایش از آدرس برای انتخاب یکی از درایه ها به کار گرفته می شود و درایه ای که حاوی کلمه داده و برچسب است. این حافظه نهان ۱۰۲۴ کلمه (معالی ۱ کیلوبایت) را در خود نگه می دارد. در سرگرس این فصل آدرس ها ۳۲ بیتی فرض شده اند. مقدار برچسب از درایه انتخاب شده توسط بخش گنجایش آدرس باید با بخش سرگرس از آدرس کلمه درخواستی تطابق داشته باشد. این حافظه نهان ۱۰۲۴ کلمه داده و هر بلاک یک کلمه ای است بنابراین ۱۰ بیت از آدرس به عنوان اندیس بکار می رود و بنابراین باید  $2^{10} = 1024$  بیت از آدرس با برچسب مقایسه شود. اگر این ۱۰ بیت مطابقت داشته و بیت اعتبار نیز روشن باشد، کلمه موجود در حافظه نهان تحویل پررونده خواهد شد و در غیر این صورت طاقان «برخ داده» است.

از آنجایی که هر بلاک  $2^m$  کلمه (معالی)  $2^{m+5}$  (بیت) است و برچسب اعتبار هر بیت یک بیت نیاز داریم. تعداد کل بیت های لازم برای این حافظه نهان

$$2^n \times (block\ size + tag\ size + valid\ field\ size)$$

$$2^n \times (2^m \times 32 + (32 - (n + m + 2)) + 1) = 2^n \times (2^m \times 32 + 31 - n - m)$$

۳

سوال: با فرض آدرس ها ۳۲ بیتی:  $n$  تعداد بیت های مورد نیاز برای بارگذاری یک حافظه  $n$  بایت با آدروی  $n$  بایت مستقیم و ظرفیت ۱۶ کلو بایت در آن هر بلوک ۴ کلمه چهار بیتی است، حافظه است؟

تعداد بیت برای بارگذاری ۴ کلمه (۳۲) در مجموع ۱۰۲۴ کلو بایت (کلمه)  $16KB = 4KW$  (کلمه)  $2^{12}$  کلو بایت

هر بلوک شامل ۴ کلمه است  $4 \times 32 = 128$  بیت کلمه یک بلوک ۱۸ بیتی (۲-۲-۱۰-۳۲) و در یک بیت کار بیت اعتبار حجم  $n$  حافظه  $n$

$$2^{10} \times (4 \times 32 + (32 - 10 - 2 - 2) + 1) = 2^{10} \times 147 = 147 Kbits = 18112 Byte$$

$$= 18,112 KByte$$

نتیجه برای یک حافظه ۱۶ کلو بیتی ۱۸,۱۱۲ کلو بایت نیاز است.

سوال: حافظه  $n$  بایت ۶۴ کلو بایت و هر بلوک شامل ۱۶ بایت. بایستی که آدرس ۱۲۰۰ قرار دارد که کوانتومی بلوک از حافظه  $n$  بایستی که آدرس خواهد بود؟ رابطه تعامد: (تعداد بلوک ها در حافظه  $n$ ) Mod (آدرس بلوک)

آدرس بلوک:  $\frac{Byte Address (بیت آدرس)}{Byte per block (تعداد بیت در هر بلوک)}$

$$\left[ \frac{Byte address}{Byte per block} \right] \times Byte per block$$

$$\left[ \frac{Byte address}{Byte per block} \right] \times Byte per block + (Byte per block - 1)$$

مثلاً با فرض ۱۶ بایت در هر بلوک، آدرس ۱۲۰۰ به بلوک ۷۵  $\frac{1200}{16} = 75$  در حافظه است و  $(75 \text{ mod } 64) = 11$  در حافظه  $n$  بایت خواهد بود. در حقیقت آیدی بلوک در هر کلمه نام آدرس در بین ۱۲۰۰ تا ۱۲۱۵ است و بالطبع تمام آیدی آدرس ها به همین بلوک تعامد می شود.

مواجهی که در هر بلوک تعداد دستور العمل  $instruction miss$  (دستال خورد):

- ۱- مقدار واقعی PC (یعنی مقدار واقعی که در برنامه من از کلمه  $PC$  را به هر بلوک حافظه ای ارسال می شود)
- ۲- به حافظه ای دستور عمل می دهد که در کلمه  $PC$  و تا تکمیل فرآیند است و این حافظه منتظر بماند
- ۳- داده مستخرج از حافظه را در خطه داده از داده منتظر بماند در حافظه  $n$  بایت  $cache$  entry تولید می کند
- ۴- محدث اجراء دستور العمل را در حافظه  $PC$  می گذارد و اگر کلمه  $PC$  را در کلمه  $PC$  قرار دهد و اگر کلمه  $PC$  را در کلمه  $PC$  قرار دهد و اگر کلمه  $PC$  را در کلمه  $PC$  قرار دهد

(۴)

دریخت محلیت نوشتن:

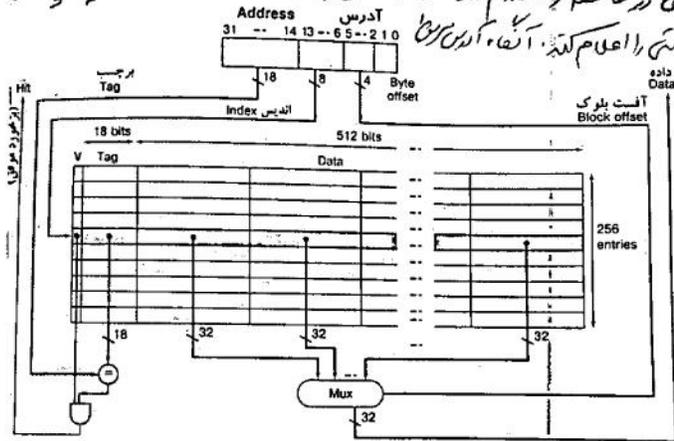
با دستورالعملی از صفحه Store داده ای را درون حافظه زبان طره data cache نوشته ایم. این حافظه  
 دیگری در حافظه اصلی تغییر کند؛ بنابراین پس از عمل نوشتن در حافظه زبان، حافظه اصلی همان مفاد را نگهداری  
 از آنجا که درون حافظه زبان از حافظه اصلی جدا شده است، بنابراین حافظه اصلی را تغییر زبان و حافظه اصلی  
 نامحتمل هستند (inconsistent) در هر دو روش مابین آنکه حافظه زبان و حافظه اصلی را جدا نگه داریم تا در هر دو  
 آن است که همیشه طره ها را در هر دو حافظه های زبان را همی بنویسیم. همین است که الگوی اصطلاحی نوشتن  
 تمام عمای write-through می گویند.

با نوشتن «write buffer» در حافظه زبان در حافظه اصلی هم می برد آن را در حافظه می طره.  
 که الگوی محتمل ترین برای نوشتن نام عمای «write-through» الگوی نوشتن «write-back» است  
 در آنجا که نوشتن و عملی در حافظه محل نوشتن همان در می شود. تعداد جدید فقط فقط درون یک حافظه از حافظه  
 این حافظه نوشته می شود و این بلوک تغییر یافته تنها زفا در حافظه سطح پایین تر از نوشتن می شود که قرار است آن  
 بلوک بلوک دیگر را نیز بنویسد. الگوی نوشتن می تواند کارایی و سرعت آن را از نو را بهبود بخشد.

۵

در این تقاضا برای عملکرد بهتر از این حافظه‌های نهان سه مدل زیر خواهد بود:

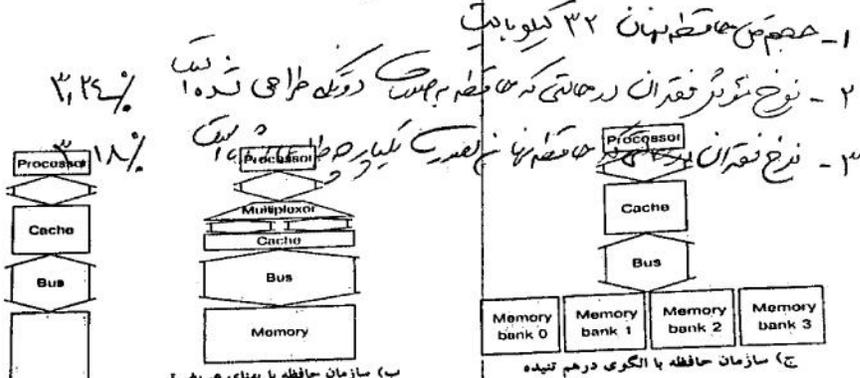
- ۱- آدرس مورد نظر بر روی حافظه نهان مستقیم بار خواهد شد و حافظه نهان آدرس را ارسال می‌کند. این آدرس را از طریق PC (نویزنده می‌نامند) و به منظور آدرس دستورالعمل مورد نیاز طریق ALU به منظور درج در حالت تک‌عملوند آماره و ارسال می‌شود.
- ۲- اگر حافظه نهان شیخ و بر روی حافظه نهان در حافظه را اعلام کند مقدار آن کلیه بررسی خطوط داده در اجتناب از پردازش خواهد بود.
- ۳- اگر حافظه نهان شیخ تعدادی حافظه را اعلام کند، آنگاه آدرس برای بار به حافظه اصلی می‌نگرد.



به محض اینکه حافظه اصلی داده را اعاده کرد آن را از روی حافظه نهان نویزده و همین بر این اساس در حالت مربوطه تک‌باردینگر آنرا می‌خواند.

شکل ۹-۵ حافظه نهان ۱۶ کیلوبایتی در پردازنده Intrinsity FastMATH دارای ۲۵۶ بلوک و هر بلوک شامل ۱۶ کلمه ۲ بایتی است. فیلد برجیب ۱۸ بیتی و فیلد اندیس ۸ بیتی است. فیلد چهاربیتی (بیت‌های دوم تا پنجم از آدرس) به عنوان اندیس یکی از کلمات از هر بلوک توسط یک ماتریس پلکسر انتخاب می‌کند.

نوع: نرخ تعداد در حافظه نهان در زیر باران برای درجه‌بندی آن و حافظه نهان یکبارگی است.



شکل ۱۱-۵ نخستین روش برای ارتقاء پهنای پهنای حافظه، افزایش پهنای فیزیکی یا منطقی سیستم حافظه است. در این شکل، پهنای اند سیستم ساده حافظه در سمت چپ به دو روش ارتقاء یافته است: استفاده از گذرگاه عرض‌تر در شکل وسط و استفاده از الگوی «درهم‌تنیده» (interleaved) در سمت راست شکل. سیستم حافظه در الگوی «درهم‌تنیده» با آرایشی عرض‌تر سازمان‌دهی و پهنای بان افزایش یافته است ولیکن گذرگاه ارتباطی بین پردازنده و حافظه عرض‌تر نشده است.

④  $\text{Multilevel cache}$  : با افزودن سطح اضافه تر به سلسله مراتب حافظه، هرچه تعداد کپی‌ها کمتر شود.

زمان  $\text{cpu}$  «  $\text{cpu Time}$  » ۱- سکل‌های سبک‌تری که  $\text{cpu}$  صرف اجرای برنامه می‌کند.  
 ۲- سکل‌های سبک‌تری که  $\text{cpu}$  در انتظار رسیدن حافظه در تعلیق می‌ماند.

$$\text{cpu Time} = (\text{cpu execution clock cycles} + \text{Memory-stall clock cycles}) \times \text{clock cycle Time.}$$

سکل‌های تعلیق ناشی از دسترس نداشتن حافظه را می‌توان به سه صورت مجموع سکل‌های تعلیق در عملیات خواندن، عملیات نوشتن و سکل‌های تعلیق در عملیات نوشتن تعریف کرد:

$$\text{Read-stall cycles} = \frac{\text{Reads}}{\text{Program}} \times \text{Read miss rate} \times \text{Read miss Penalty}$$

۱- تعداد آنتی‌ها که باید نرگفته شود.  $\text{write miss}$  ← معمولاً در همین وضعیت است که از اطلاعات ممکن نوشتن نیاز داریم تا بزرگ‌تر از آن‌ها را داشته باشیم.

۲- سکل‌های ناشی از نماندن نوشتن در  $\text{write buffer}$  در صورت نبودن حافظه در دسترس دیگر برای نوشتن، پردازنده‌ها ناچار به تعلیق می‌کند.

سکل‌های تعلیق ناشی از عملیات نوشتن به سه صورت مجموع زیرین می‌آید:

$$\text{write-stall cycles} = \left( \frac{\text{writes}}{\text{Program}} \times \text{write miss rate} \times \text{write miss penalty} \right)$$

+  $\text{write buffer stalls}$   
 نوشتن‌ها و عملیات‌های برنامه را با تعریف یک مقدار واحد برای نرخ تعداد و جمعیت تعداد به سه صورت زیر ترکیب کنیم

$$\text{Memory-stall clock cycles} = \left( \frac{\text{Memory accesses}}{\text{Program}} \times \text{Miss rate} \times \text{Miss Penalty} \right)$$

$$\Rightarrow \text{Memory-stall clock cycles} = \left( \frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Misses}}{\text{instruction}} \times \text{Miss penalty} \right)$$

⑦  $CPI = \text{طایفه‌های کارایی در هر دستورالعمل نیاز دلدیر}$

معادله کارایی حافظه زبان

مثال: نرخ فقدان در حافظه زبان دستورالعمل ۲٪

نرخ فقدان حافظه زبان داده ۴٪

نسبت تلفیق حافظه Memory Stalls دارای  $CPI = 2$

حجم معادل ۱۰۰ مگابایت با هم فریبی سرآوردگی مشاهده کرد.

مطابق دستورالعمل در زنده مانداری را ۲۴٪ فریبی کشید.

خوانندگی در متن

فقدان دستورالعمل در حساب تعداد دستورالعمل  $I =$

$$\text{instruction miss cycles} = I \times 2\% \times 100 = 2I$$

جمع (شودن)  $I$  بار خوانندگی دستورالعمل حالت

$$\text{data miss cycles} = I \times 24\% \times 4\% \times 100 = 1,224I$$

تعداد کل سکل در تلفیق نامی از حافظه عبارت است از  $2I + 1,224I = 1,226I$

یعنی از این دستورالعمل بیش از سه سکل تلفیق مشاهده داشت.

$$CPI = 2 + 1,226 = 1,228$$

نسبت زمان اجرای CPU برابر است با:

$$\frac{\text{cpu time with stalls}}{\text{cpu time with perfect cache}} = \frac{I \times CPI_{\text{stall}} \times \text{clock cycle}}{I \times CPI_{\text{perfect}} \times \text{clock cycle}} = \frac{CPI_{\text{stall}}}{CPI_{\text{perfect}}}$$

$$= \frac{1,228}{2} = 2,72$$

این اوصاف کارایی بر طایفه حافظه زبان اشاره آل  
با فریب ۲,۷۲ بهتر خواهد بود. (قریباً ۲۰٪)

برای آنکه بتوانیم زمان دسترسی به داده‌ها در درخت بزرگتر از hit به علاوه درخواستی دیگر در صورت فقدان miss، در هر یک کارایی بسنجیم و ثابت کرده‌ایم که اغلب از معیار میوم به AMAT میانگین زمان دسترسی به داده‌ها average memory access time برای تحلیل در برسی طراحی موثره بهره می‌گیرند.

$$AMAT = \text{Time for a hit} + \text{miss rate} \times \text{Miss penalty}$$

$$AMAT = \text{زمان تک‌برخ زدن موفق} + \text{نرخ فقدان} \times \text{جریمه فقدان}$$

مثال: در یک سیستم زمان دسترسی به حافظه:

مقدار AMAT را بر اساس این مشخصات زیر حساب کنید:  
 زمان هر سیکل ساعت 1ns، جریمه فقدان معادل ۲۰ سیکل ساعت، نرخ فقدان ۰.۰۵ در هر سیکل ساعت.  
 زمان دسترسی به حافظه زمان معادل ۱ سیکل ساعت.  
 فرض: جریمه فقدان معادل ۲۰ سیکل ساعت و در هر سیکل ساعت یک آیت.  
 حل: میانگین زمان دسترسی به حافظه برابر است با:

$$AMAT = \text{جریمه فقدان} \times \text{نرخ فقدان} + \text{زمان تک‌برخ زدن موفق}$$

$$= 1 + 0.05 \times 20$$

$$= 2 \text{ سیکل ساعت} = 2 \text{ نانو ثانیه}$$

نگارنده مستقیم: direct mapped ← از آدرس هر بایت در حافظه اصلی به موقعیتی واحد در حافظه بلوک بلاسازند سلسله مراتب حافظه، تنها یک نگارنده مستقیم و از قبل مشخص وجود دارد.

الگوی تمام انجمنی: Fully associative ← به بایگ از حافظه اصلی را می‌توان در هر مکان ذخیره کرد و نیاز به حافظه همان قدر دارد. در این الگوی هر بایت از حافظه اصلی می‌تواند در هر جای حافظه (entry) از حافظه نهان قرار بگیرد.

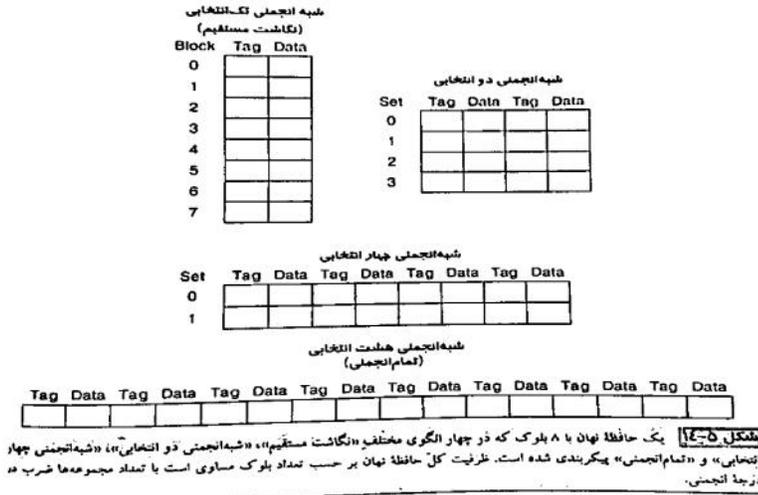
سازماندهی بایگ حافظه در حافظه نهان تمام انجمنی، تمامی داده‌ها می‌تواند در آن را جستجو کرد زیرا هر بایت ممکن است در هر مکان از حافظه نهان نهان شده باشد.

الگوی نیمه انجمنی: هر بایت را می‌توان در تعداد ثابت و مشخص از مکان حافظه نهان قرار داد.

حافظه نهان نیمه انجمنی - n استجایی ← مشخص از تعداد مجموعه (Set) در هر مجموعه طریای ظرفیتی معادل n بایت است.

در حافظه نهان مستقیم برنشانی مستقیم  
در حافظه نسبی انجمنی

④ تعداد بزرگ‌ها در حافظه نهان  $mod$  (آدرس بزرگ)  
تعداد مجموعی‌ها در حافظه نهان  $mod$  (آدرس بزرگ)



سوال: (رابطه بین تعداد درجه انجمنی در حافظه نهان):  
فرقی ندارد حافظه نهان داریم. هر یک شامل از 4 بلوک تک‌تک می‌باشد. یکی از این حافظه‌ها نهان تمام انجمنی  
دیگری شبه انجمنی دو‌التغایی و دیگری مستقیم می‌باشد. با فرض آنکه (مثلاً) بلوک‌ها 0، 4، 8، 12، 16، 20، 24، 28، 32، 36، 40، 44، 48، 52، 56، 60، 64، 68، 72، 76، 80، 84، 88، 92، 96، 100، 104، 108، 112، 116، 120، 124، 128، 132، 136، 140، 144، 148، 152، 156، 160، 164، 168، 172، 176، 180، 184، 188، 192، 196، 200، 204، 208، 212، 216، 220، 224، 228، 232، 236، 240، 244، 248، 252، 256، 260، 264، 268، 272، 276، 280، 284، 288، 292، 296، 300، 304، 308، 312، 316، 320، 324، 328، 332، 336، 340، 344، 348، 352، 356، 360، 364، 368، 372، 376، 380، 384، 388، 392، 396، 400، 404، 408، 412، 416، 420، 424، 428، 432، 436، 440، 444، 448، 452، 456، 460، 464، 468، 472، 476، 480، 484، 488، 492، 496، 500، 504، 508، 512، 516، 520، 524، 528، 532، 536، 540، 544، 548، 552، 556، 560، 564، 568، 572، 576، 580، 584، 588، 592، 596، 600، 604، 608، 612، 616، 620، 624، 628، 632، 636، 640، 644، 648، 652، 656، 660، 664، 668، 672، 676، 680، 684، 688، 692، 696، 700، 704، 708، 712، 716، 720، 724، 728، 732، 736، 740، 744، 748، 752، 756، 760، 764، 768، 772، 776، 780، 784، 788، 792، 796، 800، 804، 808، 812، 816، 820، 824، 828، 832، 836، 840، 844، 848، 852، 856، 860، 864، 868، 872، 876، 880، 884، 888، 892، 896، 900، 904، 908، 912، 916، 920، 924، 928، 932، 936، 940، 944، 948، 952، 956، 960، 964، 968، 972، 976، 980، 984، 988، 992، 996، 1000، 1004، 1008، 1012، 1016، 1020، 1024، 1028، 1032، 1036، 1040، 1044، 1048، 1052، 1056، 1060، 1064، 1068، 1072، 1076، 1080، 1084، 1088، 1092، 1096، 1100، 1104، 1108، 1112، 1116، 1120، 1124، 1128، 1132، 1136، 1140، 1144، 1148، 1152، 1156، 1160، 1164، 1168، 1172، 1176، 1180، 1184، 1188، 1192، 1196، 1200، 1204، 1208، 1212، 1216، 1220، 1224، 1228، 1232، 1236، 1240، 1244، 1248، 1252، 1256، 1260، 1264، 1268، 1272، 1276، 1280، 1284، 1288، 1292، 1296، 1300، 1304، 1308، 1312، 1316، 1320، 1324، 1328، 1332، 1336، 1340، 1344، 1348، 1352، 1356، 1360، 1364، 1368، 1372، 1376، 1380، 1384، 1388، 1392، 1396، 1400، 1404، 1408، 1412، 1416، 1420، 1424، 1428، 1432، 1436، 1440، 1444، 1448، 1452، 1456، 1460، 1464، 1468، 1472، 1476، 1480، 1484، 1488، 1492، 1496، 1500، 1504، 1508، 1512، 1516، 1520، 1524، 1528، 1532، 1536، 1540، 1544، 1548، 1552، 1556، 1560، 1564، 1568، 1572، 1576، 1580، 1584، 1588، 1592، 1596، 1600، 1604، 1608، 1612، 1616، 1620، 1624، 1628، 1632، 1636، 1640، 1644، 1648، 1652، 1656، 1660، 1664، 1668، 1672، 1676، 1680، 1684، 1688، 1692، 1696، 1700، 1704، 1708، 1712، 1716، 1720، 1724، 1728، 1732، 1736، 1740، 1744، 1748، 1752، 1756، 1760، 1764، 1768، 1772، 1776، 1780، 1784، 1788، 1792، 1796، 1800، 1804، 1808، 1812، 1816، 1820، 1824، 1828، 1832، 1836، 1840، 1844، 1848، 1852، 1856، 1860، 1864، 1868، 1872، 1876، 1880، 1884، 1888، 1892، 1896، 1900، 1904، 1908، 1912، 1916، 1920، 1924، 1928، 1932، 1936، 1940، 1944، 1948، 1952، 1956، 1960، 1964، 1968، 1972، 1976، 1980، 1984، 1988، 1992، 1996، 2000، 2004، 2008، 2012، 2016، 2020، 2024، 2028، 2032، 2036، 2040، 2044، 2048، 2052، 2056، 2060، 2064، 2068، 2072، 2076، 2080، 2084، 2088، 2092، 2096، 2100، 2104، 2108، 2112، 2116، 2120، 2124، 2128، 2132، 2136، 2140، 2144، 2148، 2152، 2156، 2160، 2164، 2168، 2172، 2176، 2180، 2184، 2188، 2192، 2196، 2200، 2204، 2208، 2212، 2216، 2220، 2224، 2228، 2232، 2236، 2240، 2244، 2248، 2252، 2256، 2260، 2264، 2268، 2272، 2276، 2280، 2284، 2288، 2292، 2296، 2300، 2304، 2308، 2312، 2316، 2320، 2324، 2328، 2332، 2336، 2340، 2344، 2348، 2352، 2356، 2360، 2364، 2368، 2372، 2376، 2380، 2384، 2388، 2392، 2396، 2400، 2404، 2408، 2412، 2416، 2420، 2424، 2428، 2432، 2436، 2440، 2444، 2448، 2452، 2456، 2460، 2464، 2468، 2472، 2476، 2480، 2484، 2488، 2492، 2496، 2500، 2504، 2508، 2512، 2516، 2520، 2524، 2528، 2532، 2536، 2540، 2544، 2548، 2552، 2556، 2560، 2564، 2568، 2572، 2576، 2580، 2584، 2588، 2592، 2596، 2600، 2604، 2608، 2612، 2616، 2620، 2624، 2628، 2632، 2636، 2640، 2644، 2648، 2652، 2656، 2660، 2664، 2668، 2672، 2676، 2680، 2684، 2688، 2692، 2696، 2700، 2704، 2708، 2712، 2716، 2720، 2724، 2728، 2732، 2736، 2740، 2744، 2748، 2752، 2756، 2760، 2764، 2768، 2772، 2776، 2780، 2784، 2788، 2792، 2796، 2800، 2804، 2808، 2812، 2816، 2820، 2824، 2828، 2832، 2836، 2840، 2844، 2848، 2852، 2856، 2860، 2864، 2868، 2872، 2876، 2880، 2884، 2888، 2892، 2896، 2900، 2904، 2908، 2912، 2916، 2920، 2924، 2928، 2932، 2936، 2940، 2944، 2948، 2952، 2956، 2960، 2964، 2968، 2972، 2976، 2980، 2984، 2988، 2992، 2996، 3000، 3004، 3008، 3012، 3016، 3020، 3024، 3028، 3032، 3036، 3040، 3044، 3048، 3052، 3056، 3060، 3064، 3068، 3072، 3076، 3080، 3084، 3088، 3092، 3096، 3100، 3104، 3108، 3112، 3116، 3120، 3124، 3128، 3132، 3136، 3140، 3144، 3148، 3152، 3156، 3160، 3164، 3168، 3172، 3176، 3180، 3184، 3188، 3192، 3196، 3200، 3204، 3208، 3212، 3216، 3220، 3224، 3228، 3232، 3236، 3240، 3244، 3248، 3252، 3256، 3260، 3264، 3268، 3272، 3276، 3280، 3284، 3288، 3292، 3296، 3300، 3304، 3308، 3312، 3316، 3320، 3324، 3328، 3332، 3336، 3340، 3344، 3348، 3352، 3356، 3360، 3364، 3368، 3372، 3376، 3380، 3384، 3388، 3392، 3396، 3400، 3404، 3408، 3412، 3416، 3420، 3424، 3428، 3432، 3436، 3440، 3444، 3448، 3452، 3456، 3460، 3464، 3468، 3472، 3476، 3480، 3484، 3488، 3492، 3496، 3500، 3504، 3508، 3512، 3516، 3520، 3524، 3528، 3532، 3536، 3540، 3544، 3548، 3552، 3556، 3560، 3564، 3568، 3572، 3576، 3580، 3584، 3588، 3592، 3596، 3600، 3604، 3608، 3612، 3616، 3620، 3624، 3628، 3632، 3636، 3640، 3644، 3648، 3652، 3656، 3660، 3664، 3668، 3672، 3676، 3680، 3684، 3688، 3692، 3696، 3700، 3704، 3708، 3712، 3716، 3720، 3724، 3728، 3732، 3736، 3740، 3744، 3748، 3752، 3756، 3760، 3764، 3768، 3772، 3776، 3780، 3784، 3788، 3792، 3796، 3800، 3804، 3808، 3812، 3816، 3820، 3824، 3828، 3832، 3836، 3840، 3844، 3848، 3852، 3856، 3860، 3864، 3868، 3872، 3876، 3880، 3884، 3888، 3892، 3896، 3900، 3904، 3908، 3912، 3916، 3920، 3924، 3928، 3932، 3936، 3940، 3944، 3948، 3952، 3956، 3960، 3964، 3968، 3972، 3976، 3980، 3984، 3988، 3992، 3996، 4000، 4004، 4008، 4012، 4016، 4020، 4024، 4028، 4032، 4036، 4040، 4044، 4048، 4052، 4056، 4060، 4064، 4068، 4072، 4076، 4080، 4084، 4088، 4092، 4096، 4100، 4104، 4108، 4112، 4116، 4120، 4124، 4128، 4132، 4136، 4140، 4144، 4148، 4152، 4156، 4160، 4164، 4168، 4172، 4176، 4180، 4184، 4188، 4192، 4196، 4200، 4204، 4208، 4212، 4216، 4220، 4224، 4228، 4232، 4236، 4240، 4244، 4248، 4252، 4256، 4260، 4264، 4268، 4272، 4276، 4280، 4284، 4288، 4292، 4296، 4300، 4304، 4308، 4312، 4316، 4320، 4324، 4328، 4332، 4336، 4340، 4344، 4348، 4352، 4356، 4360، 4364، 4368، 4372، 4376، 4380، 4384، 4388، 4392، 4396، 4400، 4404، 4408، 4412، 4416، 4420، 4424، 4428، 4432، 4436، 4440، 4444، 4448، 4452، 4456، 4460، 4464، 4468، 4472، 4476، 4480، 4484، 4488، 4492، 4496، 4500، 4504، 4508، 4512، 4516، 4520، 4524، 4528، 4532، 4536، 4540، 4544، 4548، 4552، 4556، 4560، 4564، 4568، 4572، 4576، 4580، 4584، 4588، 4592، 4596، 4600، 4604، 4608، 4612، 4616، 4620، 4624، 4628، 4632، 4636، 4640، 4644، 4648، 4652، 4656، 4660، 4664، 4668، 4672، 4676، 4680، 4684، 4688، 4692، 4696، 4700، 4704، 4708، 4712، 4716، 4720، 4724، 4728، 4732، 4736، 4740، 4744، 4748، 4752، 4756، 4760، 4764، 4768، 4772، 4776، 4780، 4784، 4788، 4792، 4796، 4800، 4804، 4808، 4812، 4816، 4820، 4824، 4828، 4832، 4836، 4840، 4844، 4848، 4852، 4856، 4860، 4864، 4868، 4872، 4876، 4880، 4884، 4888، 4892، 4896، 4900، 4904، 4908، 4912، 4916، 4920، 4924، 4928، 4932، 4936، 4940، 4944، 4948، 4952، 4956، 4960، 4964، 4968، 4972، 4976، 4980، 4984، 4988، 4992، 4996، 5000، 5004، 5008، 5012، 5016، 5020، 5024، 5028، 5032، 5036، 5040، 5044، 5048، 5052، 5056، 5060، 5064، 5068، 5072، 5076، 5080، 5084، 5088، 5092، 5096، 5100، 5104، 5108، 5112، 5116، 5120، 5124، 5128، 5132، 5136، 5140، 5144، 5148، 5152، 5156، 5160، 5164، 5168، 5172، 5176، 5180، 5184، 5188، 5192، 5196، 5200، 5204، 5208، 5212، 5216، 5220، 5224، 5228، 5232، 5236، 5240، 5244، 5248، 5252، 5256، 5260، 5264، 5268، 5272، 5276، 5280، 5284، 5288، 5292، 5296، 5300، 5304، 5308، 5312، 5316، 5320، 5324، 5328، 5332، 5336، 5340، 5344، 5348، 5352، 5356، 5360، 5364، 5368، 5372، 5376، 5380، 5384، 5388، 5392، 5396، 5400، 5404، 5408، 5412، 5416، 5420، 5424، 5428، 5432، 5436، 5440، 5444، 5448، 5452، 5456، 5460، 5464، 5468، 5472، 5476، 5480، 5484، 5488، 5492، 5496، 5500، 5504، 5508، 5512، 5516، 5520، 5524، 5528، 5532، 5536، 5540، 5544، 5548، 5552، 5556، 5560، 5564، 5568، 5572، 5576، 5580، 5584، 5588، 5592، 5596، 5600، 5604، 5608، 5612، 5616، 5620، 5624، 5628، 5632، 5636، 5640، 5644، 5648، 5652، 5656، 5660، 5664، 5668، 5672، 5676، 5680، 5684، 5688، 5692، 5696، 5700، 5704، 5708، 5712، 5716، 5720، 5724، 5728، 5732، 5736، 5740، 5744، 5748، 5752، 5756، 5760، 5764، 5768، 5772، 5776، 5780، 5784، 5788، 5792، 5796، 5800، 5804، 5808، 5812، 5816، 5820، 5824، 5828، 5832، 5836، 5840، 5844، 5848، 5852، 5856، 5860، 5864، 5868، 5872، 5876، 5880، 5884، 5888، 5892، 5896، 5900، 5904، 5908، 5912، 5916، 5920، 5924، 5928، 5932، 5936، 5940، 5944، 5948، 5952، 5956، 5960، 5964، 5968، 5972، 5976، 5980، 5984، 5988، 5992، 5996، 6000، 6004، 6008، 6012، 6016، 6020، 6024، 6028، 6032، 6036، 6040، 6044، 6048، 6052، 6056، 6060، 6064، 6068، 6072، 6076، 6080، 6084، 6088، 6092، 6096، 6100، 6104، 6108، 6112، 6116، 6120، 6124، 6128، 6132، 6136، 6140، 6144، 6148، 6152، 6156، 6160، 6164، 6168، 6172، 6176، 6180، 6184، 6188، 6192، 6196، 6200، 6204، 6208، 6212، 6216، 6220، 6224، 6228، 6232، 6236، 6240، 6244، 6248، 6252، 6256، 6260، 6264، 6268، 6272، 6276، 6280، 6284، 6288، 6292، 6296، 6300، 6304، 6308، 6312، 6316، 6320، 6324، 6328، 6332، 6336، 6340، 6344، 6348، 6352، 6356، 6360، 6364، 6368، 6372، 6376، 6380، 6384، 6388، 6392، 6396، 6400، 6404، 6408، 6412، 6416، 6420، 6424، 6428، 6432، 6436، 6440، 6444، 6448، 6452، 6456، 6460، 6464، 6468، 6472، 6476، 6480، 6484، 6488، 6492، 6496، 6500، 6504، 6508، 6512، 6516، 6520، 6524، 6528، 6532، 6536، 6540، 6544، 6548، 6552، 6556، 6560، 6564، 6568، 6572، 6576، 6580، 6584، 6588، 6592، 6596، 6600، 6604، 6608، 6612، 6616، 6620، 6624، 6628، 6632، 6636، 6640، 6644، 6648، 6652، 6656، 6660، 6664، 6668، 6672، 6676، 6680، 6684، 6688، 6692، 6696، 6700، 6704، 6708، 6712، 6716، 6720، 6724، 6728، 6732، 6736، 6740، 6744، 6748، 6752، 6756، 6760، 6764، 6768، 6772، 6776، 6780، 6784، 6788، 6792، 6796، 6800، 6804، 6808، 6812، 6816، 6820، 6824، 6828، 6832، 6836، 6840، 6844، 6848، 6852، 6856، 6860، 6864، 6868، 6872، 6876، 6880، 6884، 6888، 6892، 6896، 6900، 6904، 6908، 6912، 6916، 6920، 6924، 6928، 6932، 6936، 6940، 6944، 6948، 6952، 6956، 6960، 6964، 6968، 6972، 6976، 6980، 6984، 6988، 6992، 6996، 7000، 7004، 7008، 7012، 7016، 7020، 7024، 7028، 7032، 7036، 7040، 7044، 7048، 7052، 7056، 7060، 7064، 7068، 7072، 7076، 7080، 7084، 7088، 7092، 7096، 7100، 7104، 7108، 7112، 7116، 7120، 7124، 7128، 7132، 7136، 7140، 7144، 7148، 7152، 7156، 7160، 7164، 7168، 7172، 7176، 7180، 7184، 7188، 7192، 7196، 7200، 7204، 7208، 7212، 7216، 7220، 7224، 7228، 7232، 7236، 7240، 7244، 7248، 7252، 7256، 7260، 7264، 7268، 7272، 7276، 7280، 7284، 7288، 7292، 7296، 7300، 7304، 7308، 7312، 7316، 7320، 7324، 7328، 7332، 7336، 7340، 7344، 7348، 7352، 7356، 7360، 7364، 7368، 7372، 7376، 7380، 7384، 7388، 7392، 7396، 7400، 7404، 7408، 7412، 7416، 7420، 7424، 7428، 7432، 7436، 7440، 7444، 7448، 7452، 7456، 7460، 7464، 7468، 7472، 7476، 7480، 7484، 7488، 7492، 7496، 7500، 7504، 7508، 7512، 7516، 7520، 7524، 7528، 7532، 7536، 7540، 7544، 7548، 7552، 7556، 7560، 7564، 7568، 7572، 7576، 7580، 7584، 7588، 7592، 7596، 7600، 7604، 7608، 7612، 7616، 7620، 7624، 7628، 7632، 7636، 7640، 7644، 7648، 7652، 7



مثال: اندازه برچسب و رجیم انجمنی در حافظه نشان  
 با فرض بر این که هر بیت ۱۲ بیت (۲<sup>۴</sup>) ظرفیت دارد از این روز مجموع ۳۲ بیت ۲۸ بیت (۲<sup>۴</sup>-۲) است  
 در تعداد بیت های برچسب را برای حافظه نشان، تفاوت مستقیم، انجمنی در انتخابی و انجمنی به انتخابی و تمام  
 انجمنی می باشد.  
 حل: از آنجایی که هر بیت ۱۲ بیت (۲<sup>۴</sup>) ظرفیت دارد از این روز مجموع ۳۲ بیت ۲۸ بیت (۲<sup>۴</sup>-۲) است  
 برای اندازه های اندیس و برچسب باقی می ماند.  
 حافظه نشان تفاوت مستقیم تعداد مجموع و اندازه برچسبها یکسان است.  $\log_2(4k) = 12$  بنابراین اندازه های برچسبها  
 مجموع من بیت های برچسب  $16 \times 4k = (28-12) \times 4k$  معادل 64 کیلوبیت خواهد بود.

برای حافظه نشان نسبه انجمنی با دو انتخاب (2-way) جمعاً ۲۰۶۸ مجموع خواهیم داشت. و تعداد بیت های برچسب ۴۸ بیت  
 $(28-11) \times 2 \times 4k = 24 \times 4k$   
 حافظه نشان نسبه انجمنی با چهار انتخاب (4-way) تعداد کل مجموع ۱۰۲۴ است. تعداد بیت های برچسب معادل  
 $(28-10) \times 4 \times 4k = 72 \times 4k = 72k$   
 حافظه نشان تمام انجمنی انتخاب مجموع ۵۹۶ بیت و هر یک و در هر دو و در هر دو برچسب ۲۸ بیت است.  
 بنابراین تعداد کل بیت های برچسب معادل  $28 \times 4 \times 4k = 112k$  بیت خواهد بود.

مثال: کارایی حافظه نشان غیرسطحی  
 برآوردی از زمان در دسترس بودن (hit) برای میسر. دارای مقدار معیاری  $CPI = 1.0$   
 نرخ سفارش است نیز  $EGHZ$  زمان دسترسی به حافظه اصلی (شامل تمام هزینه های ناشی از دسترسی به حافظه)  $100ns$   
 مجموع نرخ فقدان miss rate به ازای هر دستور العمل ۲٪  
 برآوردی از زمان در دسترس بودن  $L2$  cache: زمان دسترسی  $5ns$   
 نرخ فقدان ۵٪ کاهش باید به آنگاه برآوردی از زمان در دسترس بودن سرخط خواهد بود.

$$N \times T + 0.02 N \times 400 T = 1N T + 8N T = 9N T$$

$$T = \frac{1}{4 \times 10^9} = \frac{10^9}{4} = \frac{1}{4} \times 10^9$$

$$Tr = \frac{100 \times 10^{-9}}{10^9/4} = 400$$

$$Total CPI = Base CPI + memory-stall \text{ cycles per instruction} \Rightarrow Total CPI = 1.0 + MemoryStall = 1.0 + 2\% \times 400 = 9$$

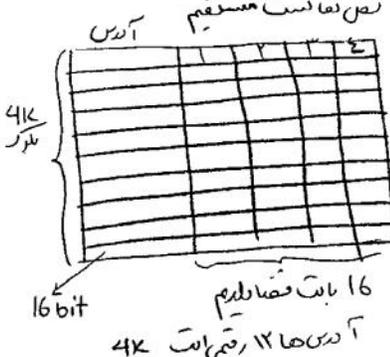
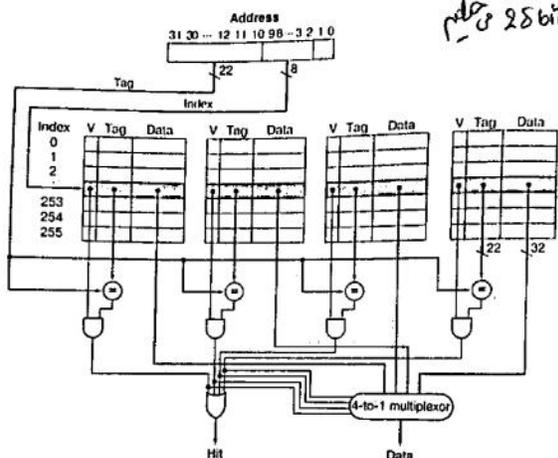
$$N T + 0.02 N \times 20 T + 0.005 \times 400 T \Rightarrow N T + 0.4 N T + 2 N T = 3.4 N T$$

$$Total CPI = Base CPI + Primary Stall per instruction + Secondary Stall per instruction \Rightarrow 1 + 2\% + 20 + 0.5\% \times 400 = 14 + 2 = 16$$

$$\frac{9}{16} = 0.5625$$

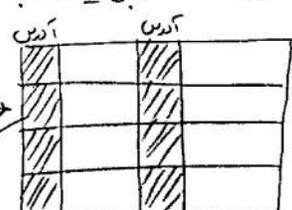
۱۳

در حالت مستقیم یک بیت کده 16 بیتی  
در حالت تمام احتمالی 4096 مقایسه کننده 28 بیت می‌دهیم  
تصویر حالت مستقیم



شکل ۵-۱۷: پادسازی یک حافظه نهان سه‌التمسی چهار التمسی نیازمند ۴ مقایسه کننده و یک مالتی پلکسر ۴ به ۱ است. مقایسه کننده‌ها تعیین می‌کنند که کدام یک از اعضای مجموعه انتخاب شده یا نپذیرد. مطابقت دارند (البته در صورت حضور) خروجی مقایسه کننده‌ها برای کنترل مالتی پلکسر مورد استفاده قرار می‌گیرند تا این مالتی پلکسر از بین یکی از ۴ عضو هر بلوک یکی را به خروجی هدایت کند. خروجی Hit تعیین کننده آن است که آیا داده مورد نظر در حافظه نهان وجود داشته است یا خیر. مشاغلر ظاهر کننده در خروجی مالتی پلکسر در صورت فعال بودن این خط قابل استفاده توسط پردازنده هستند.

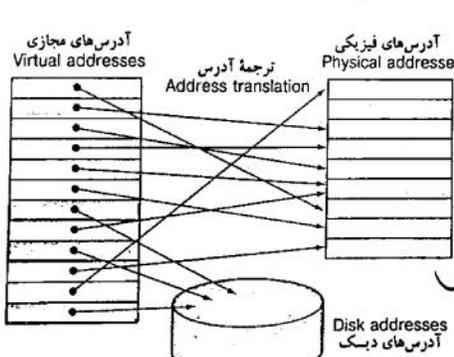
تصویر حالت احتمالی ۲ انتخابی



انتخابی ۲ انتخابی ← مقایسه کننده ۱۸ بیتی

حافظه مجازی

صحنه‌ها و نقشه‌ها می‌بینند اما  
۴۰۹۶ بیتی ۱۶۱۲۲۱ می‌بینند



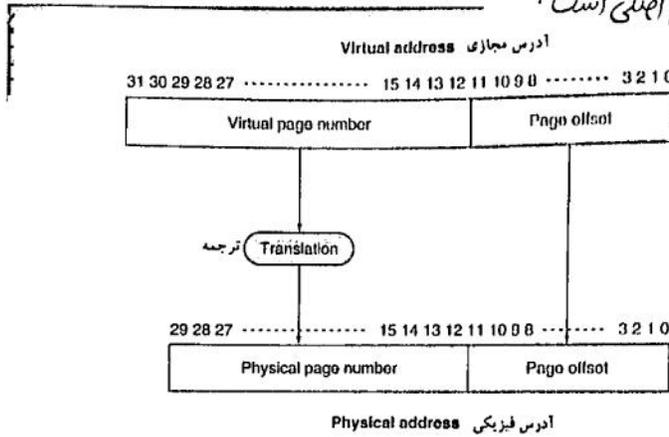
تبدیل از حافظه مجازی به فیزیکی  
همانند  
حواکه داده‌ها در حافظه مجازی نیستند  
به مقدار آن اطلاعات نقصان صفحه  
Page fault  
فرستادن

برای رجوع به حافظه مجازی یک آدرس مجازی  
تولید می‌کنند و این آدرس به کمک ترکیبی  
از سخت افزار و نرم افزار  
به آدرس فیزیکی ترجمه  
می‌کنند

شکل ۵-۱۹: در حافظه مجازی بلوک‌های حافظه (موسوم به صفحات) از مجموعه‌ای از آدرس‌های مجازی به مجموعه‌ای دیگر از آدرس‌های فیزیکی نگاشته می‌شوند. آدرس‌هایی که پردازنده تولید می‌کند مجازی هستند در حالی که برای دستیابی به مقادیر واقعی در حافظه به آدرس‌های فیزیکی نیاز است. چه حافظه مجازی و چه حافظه فیزیکی به قطعاتی موسوم به صفحه (page) تکه تکه می‌شوند و هر صفحه مجازی به یک صفحه فیزیکی نگاشته خواهد شد. البته این امکان وجود دارد که یک صفحه مجازی در حافظه اصلی موجود نباشد. بدین ترتیب به هیچ صفحه فیزیکی نگاشته نشده باشد. در این صورت باید بر روی دیسک سخت به دنبال آن گشت. یک صفحه فیزیکی واحد ممکن است با دو یا چند آدرس مجازی متفاوت بین چند برنامه به اشتراک گذاشته شود.

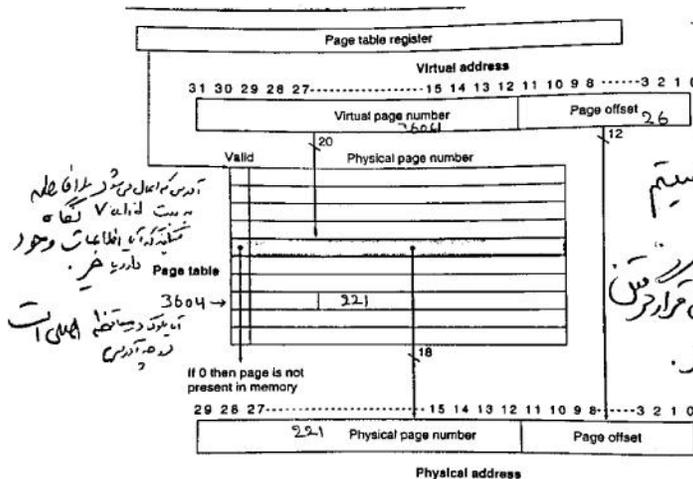
در صفحه مجازی آدرس به دو بخش شماره صفحه مجازی virtual page number و آخرت صفحه page offset تقسیم می‌شود.  
شماره صفحه فیزیکی به بخش پردازش از آدرس فیزیکی را تشکیل می‌دهد.  
مقدار آخرت صفحه به هیچ فیزیکی بخش کم‌پردازش از آدرس فیزیکی را تشکیل می‌دهد.

در سیستم‌های حافظه مجازی برای تعیین موقعیت و یافتن یک صفحه در حافظه اصلی (زنجیره‌ای برای) اندیس‌های مجازی ساخته می‌شود. همین سیستم‌های اصطلاحاً در جدول صفحات Page table نامیده می‌شوند و محل نگهداری آن در همان حافظه اصلی است.



**شکل ۵-۲۰** نگاشت آدرس‌های مجازی به آدرس‌های فیزیکی. اندازه هر صفحه ۱ کیلو بایت (۲<sup>۱۰</sup> بایت) و تعداد صفحات فیزیکی برابر در حافظه اصلی ۲<sup>۱۸</sup> صفحه است زیرا از یک آدرس ۳۲ بیتی فقط ۱۸ بیت برای آدرس‌دهی صفحات فیزیکی باقی می‌ماند. با این اوصاف حافظه اصلی می‌تواند حداکثر یک گیگابایت ظرفیت داشته باشد در حالی که فضای آدرس‌دهی مجازی ۱ گیگابایت است.

جدول صفحات اندیس‌های مجازی و فیلد شماره صفحه در طبقه آدرس مجازی به عنوان اندیس برای مراجعه به این جدول نگار گرفته می‌شود. و شماره فیزیکی صفحه منطبقاً را در حافظه اصلی مشخص می‌کند.



آدرس‌های مجازی در طرف اول به بیت‌های ۷ نگاه کنید که آن‌ها اطلاعات وجود دارند هر

3604 →

آدرس‌های مجازی در این جدول

نصف صفحات به هر دو بیت اعتبار برای یک صفحه مجازی خاص باقی‌مانده است. در ضمن شرایطی مانند تبدیل اعداد سیستم عامل سرور شود.

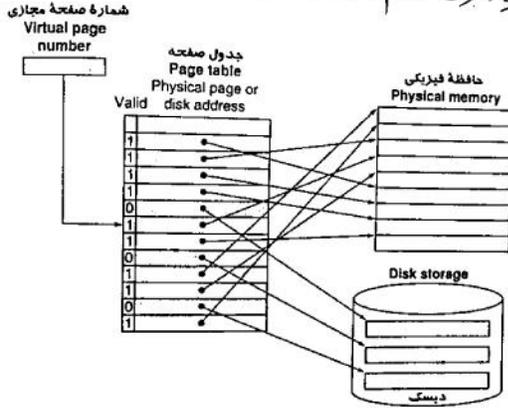
آدرس مجازی به خودی خود نمی‌تواند مستقیماً عمل کرد. صفحات را بر روی دست‌نویس مشخص کند.

**شکل ۵-۲۱** جدول صفحات (page table) بر اساس شماره صفحه مجازی، ایندکس‌دهی شده و مورد مراجعه قرار می‌گیرد تا بدین طریق بخش پروازش از آدرس فیزیکی متناظر به دست آید. در این شکل آدرس ۳۲ بیتی و اندازه صفحات ۱ کیلو بایت فرض شده‌اند. آدرس شروع جدول صفحات توسط اشاره گر «حیات جدول صفحه» (page table register) تعیین می‌شود. تعداد درایه‌های جدول صفحه ۲<sup>۱۰</sup> یعنی حدود ۱ میلیون درایه است. جهت اعتبار «Valid bit» مشخص می‌کند که آیا نگاشت مربوطه معتبر است یا باید صفحه مجازی را بر روی دیسک جستجو کرد. هر چند در اینجا هر درایه ۱۹ بیت است ولی عمدتاً برای سهولت ایندکس‌دهی به ۲۲ بیت گنجانده می‌شود و از بیت‌های مازاد برای حفاظت مثل حفاظت (protection) استفاده می‌شود.

برای مشخص کردن موقعیت جدول صفحات در حافظه اصلی، سیستم‌ها از یک بیت خاص (برای اشاره به نقطه شروع) در جدول صفحات Page table استفاده می‌کنند. همین‌طور در اختیار می‌گذارند؛ همین‌طور با بیت جدول صفحات Page table register.

۱۴

از آنجا که همیشه نمی‌دانیم که یک صفحه در حافظه فیزیکی جایگزین (replace) شود یا نه، سیستم عامل معمولاً وقتی بر روی یک آدرس در حافظه فیزیکی تمام صفحات مورد استفاده آن بر روی یک آدرس جایگزین می‌شود.



به این بخش از دیسک اصطلاحاً فضای مجازی Space گفته می‌شود.

همین سیستم عامل در لحظه ای که در این مکانها داده‌ای را وجود می‌آورد تا در آن نشانی دقیق ذخیره هر صفحه عملی بر روی دیسک ثبت و ضبط شود.

شکل ۵-۲۱ جدول صفحه: یکایک صفحات در حافظه مجازی را به صفحات حافظه اصلی یا صفحات ذخیره شده بر روی دیسک (به عنوان سطح بندی در سلسله مراتب حافظه) می‌نگارد. بخش شماره صفحه مجازی «در بیان هر آدرس به عنوان آدرس به جدول صفحه اعمال شده و سپس در درایه متناظر، هیت اعتبار» آزمایش می‌شود. اگر این بیت فعال بود درایه متناظر مستقیماً شماره صفحه فیزیکی را در اختیار می‌گذارد ولیکن در صورت خاموش بودن هیت اعتبار، آن صفحه در آدرس مشخصی از دیسک سخت قرار گرفته است. در بسیاری از سیستمها آدرس صفحات فیزیکی حافظه و آدرس صفحات دیسک علی‌رغم آنکه منطقاً یک جدول واحد هستند ولیکن در دو ساختمان داده مجزا نگهداری می‌شوند.

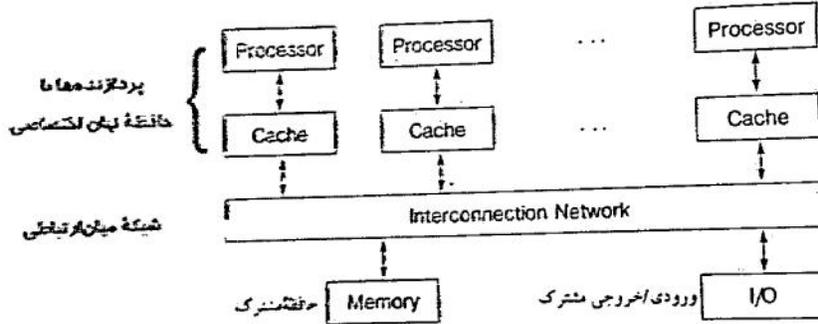
سیستم عامل همیشه سعی می‌کند که از طرف آن مشخص کند که هر یک از برنامه‌ها هر یک از آدرس‌های مجازی از کدامین صفحه فیزیکی در حافظه اصلی استفاده کرده‌اند. مابقی دارند فضای لغت Page fault اگر تمام صفحات حافظه اصلی را اشغال باشند سیستم عامل موظف است صفحه‌ای را برای جایگزینی کردن در بر روی انداختن از حافظه اصلی ذخیره کند. از آنجا که کامل داریم تعداد رخداد های لغت صفحه به کمترین ممکن برسد سیستم عامل می‌کوشد تا صفحه‌ای را برای جایگزینی انتخاب کند که کمترین هم‌پوشانی داشته باشد. آن نیازی نخواهد بود.

مخوبی نوشتن در حافظه مجازی:

در حافظه مجازی نوشتن در سطح فیزیکی از سلسله مراتب حافظه (یعنی دست‌نویس) مدنظرمانند از سلسله مراتب سلسله مراتب عملی است. به طرز عمده‌ای نامر از این رو تعبیه می‌شود تا نوشتن برای آنکه سیستم بتواند عمل نوشتن را به صورت تمام عیار انجام دهد. کاملاً ناممکن است. در مقابل سیستم حافظه باید از روش باز نویسی استفاده کند. write-back نام این نحوه داده‌ها فقط در حافظه اصلی نوشته شده و بعداً در حافظه فیزیکی ذخیره می‌شود. آن صفحه بر روی دیسک باز نویسی می‌شود.

## فصل هفتم

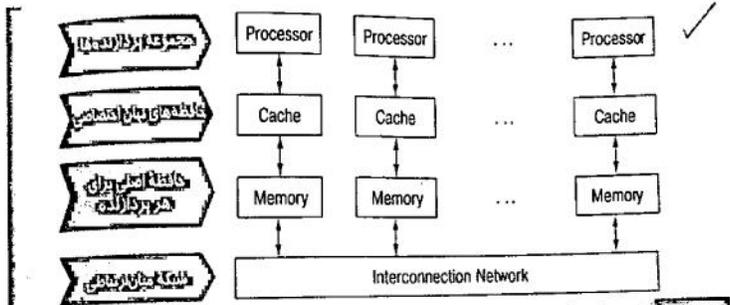
① یک چندپردازنده با حافظه اشتراکی (Shared memory multiprocessor) SMP به اشتراک گذاشتن یک فضای آدرس دهی مشترک بین تمام پردازنده‌ها، از یک برنامه نویسان حافظه ای بسیار آسان است. حتی اگر هر چند با همین تعریفی ناهمخوانی چند پردازنده با فضای آدرس مشترک گویا در تقویت ترسیم نظر می رسد



شکل ۷-۲ سازمان کلاسیک «چند پردازنده» با فضای حافظه مشترک.

۱- زمان دسترسی به حافظه اصلی فارغ از آنکه کدام پردازنده کش‌های دسترسی داشته و کدامی کلیم از حافظه را درخواست کرده است و این وکتورهای نامی می ماند. به همین واسطه برای محوما چند پردازنده در UMA گفته می شود.

۲- برخی از دسترسی‌ها و مراعات به حافظه است. آنکه کدام پردازنده کش‌های حافظه را درخواست داده است بسید سرفتر از تقسیم قدرت می گردد. همین واسطه‌های چند پردازنده در UMA نام گرفته است.



شکل ۷-۳ سازمانی کلاسیک از یک چند پردازنده که در آن هر یک از پردازنده‌ها از فضای آدرس اختصاصی خودشان بهره می گیرند و از طریق یک شبکه میان ارتباطی (interconnection network) با یکدیگر به «مبادله پیام» می پردازند. این ساختار به طور سنتی «چند پردازنده مبتنی بر مبادله پیام» نامیده می شود. دقت کنید که برخلاف ساختار SMP در شکل ۷-۲ شبکه میان ارتباطی بین حافظه نهان و حافظه اصلی قوی ندارد بلکه بین گروه‌های مشتمل بر پردازنده حافظه «ایفای نقش» می کند.

مثال بهره می حافظه: در پردازنده اشتراکی SMP راه محض ۲۰ گنجایت از حافظه اصلی است در ساین پنج کامپیوتر خوشه‌ای clustered که هر کدام ۸ گنجایت حافظه دارند و سطح عملی به یک گنجایت نقدی همان حافظه را اتقار می آید. در نظر سید: در ساین به پردازنده حافظه ۲۰۰ در ساین به بران می آید

$$\frac{20-1}{5 \times (2-1)} = \frac{19}{15} = 1,25$$